Bio-Inspired, Odor-Based Navigation

THESIS

Maynard John Porter III, Captain, USAF

AFIT/GE/ENG/06-48

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GE/ENG/06-48

Bio-Inspired, Odor-Based Navigation

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Maynard John Porter III, B.S.E.E.

Captain, USAF

March 2006

AFIT/GE/ENG/06-48

Bio-Inspired, Odor-Based Navigation

Maynard John Porter III, B.S.E.E.
Captain, USAF

Approved:

LtCol J.R. Vasquez, PhD Thesis Advisor	Date
Dr. G.B. Lamont Committee Member	Date
Dr. M.M. Miller Committee Member	Date

Abstract

The ability of many insects, especially moths, to locate either food or a member of the opposite sex, by tracking a wind-borne plume of odor molecules, is an amazing reality. Numerous scenarios exist where having this capability embedded into ground-based or aerial vehicles would be invaluable. The main crux of this thesis investigation is the development of a navigation algorithm which gives a UAV the ability to track a chemical plume to its source. Inspiration from the male moth's, in particular Manduca sexta (Tobacco Hornworm moth), ability to successfully track a female's pheromone plume was used in the design of both 2-D and 3-D navigation algorithms. The algorithms were developed to guide autonomous vehicles to a source generating an odor/chemical plume, using only the odor/chemical information provided by the plume.

The algorithms were implemented using a variety of fuzzy controllers and ad hoc engineering approaches. The fuzzy controller, critical to the success of both algorithms, was developed to estimate the location of a vehicle relative to the plume: coming into the plume, in the plume, exiting the plume, or out of the plume. Analysis of plume detections within a short-term memory bank provided the basis for this controller.

To test these algorithms, 2-D and 3-D simulation environments were developed. Both environments contain vehicle dynamics, sensor, and time-varying plume models. The more complex 3-D environment included a 6-degree of freedom, nonlinear aircraft model designed to represent a small UAV. These simulations were executed for varying plume structures and densities, ensuring robustness of the navigation algorithms. Both algorithms are promising. The 2-D algorithm had a 60% to 90% success rate in reaching the source while certain versions of 3-D algorithm had success rates from 50% to 100%.

Acknowledgements

I would like to express my sincere appreciation to my thesis advisor Lt Col Juan Vasquez, for his guidance and support through this endeavor. Without his insight and guidance, this effort would not have been nearly as successful. I would also like to thank Dr. Mark Willis of Case Western Reserve University, as his in-depth knowledge on the behavior of Manduca Sexta was invaluable to the development of the odor-based navigation algorithms designed in this thesis effort. Without the aircraft dynamics expertise of Dr. Brad Liebst, the 3-D simulations developed for this research would never have materialized. For that, my gratitude.

I am forever indebted to my parents. Without their tireless support and immense sacrifice, I would not have had the fortitude to complete this thesis. To my sisters, whose words of encouragement have helped me through the hardest of times, thank you. I also owe a great deal to my uncle who, during his too-short like offered me nothing but support. A special thank you goes to my wife who not only served as an extra set of eyes in editing this thesis, but who cheered me on, helping me to see the light at the end of the tunnel. Finally, to my daughter, who made this whole endeavor worth while. Without your smile I would have never succeeded.

Maynard John Porter III

Table of Contents

Page

List of Figures

Figure Page

List of Tables

List of Abbreviations

Abbreviation		Page
MSexta	Manduca Sexta	1-2
UAV	Unmanned Aerial Vehicle	1-3
AUV	Autonomous Under Water Vehicle	2-27

Bio-Inspired, Odor-Based Navigation

I. Introduction

Humans have, for centuries, endeavored to mimic Nature's abilities. What better source to draw from than creatures which have evolved over millions of years, to become the best at what they do. Reverse engineering these extraordinary abilities, such as odor-based navigation, could lead to important advancements in many areas, navigation being just one of them. Consider a burning building. A robotic dog, without the fear of fire and immune to the toxicity of smoke, is dispatched to search for potential casualties using sight, sound, and smell. Another scenario: the military is hunting a terrorist in a remote area. A micro-unmanned aerial vehicle "trained" on a certain smell (human scent) flies just meters off the ground, searching autonomously and efficiently. Incorporating such ability in today's search and rescue and other military operations would be invaluable. In the future sensors that can detect finite levels of chemicals will be small and sensitive enough to be used on vehicles and robots. However, once such sensing ability is achieved, the question becomes how to exploit it or use it as a navigation aid. One must develop a navigation scheme incorporating the sense of smell. Developing such a 3-D navigation algorithm for small scale unmanned aerial vehicle is the primary goal of this thesis effort.

1.1 Biological Inspiration

Much work has been done in the biological and behavioral realm observing animals that depend on their sense of smell or the ability to detect airborne chemicals for survival or reproduction. Lobsters, certain seabirds, cockroaches, flies, and moths each depend on this ability to find food or a mate [2, 12, 16, 43, 47, 50]. At first glance, one might think that securing sustenance would be paramount; however, successful reproduction is the ultimate goal. Moths, in particular Manduca sexta (Tobacco

Hornworm moth) (MSexta), have been studied for more than two decades [5]. This work, together with decades of research on plume tracking behavior in other moth species has resulted in the development of hypotheses explaining how a flying organism might track an odor plume carried by a dynamically changing wind. A key component of the MSexta research is the wind tunnel experiments described in [6, 50]. The research on this behavior in MSexta and other flying insects has almost always focused on the horizontal maneuvers since moths appear to maintain level flight once detecting the pheromone plume. This prompted the need for the secondary objective of this thesis effort: developing a 2-D navigation algorithm mimicking the moth's capabilities found in the wind tunnel experiments. The successful 2-D algorithms were then used as the starting point for the development of the objective of developing an algorithm that can control plume tracking behavior in a 3-D environment.

1.2 Previous and/or Related Research

Many 2-D computer simulations and robotic implementations of odor based navigation schemes have been developed over the last 20 years [5, 12, 16, 21, 29, 43]. These provide insight into the horizontal component of the 3-D navigation algorithm but do not aid in the design of the vertical component. The horizontal methodology for navigating a chemical plume presented in this thesis research has a unique methodology; using a short term memory to help make maneuver decisions. The hypothesis behind this methodology is discussed in Chapter II and the technique itself is described in Chapter III. There is a negligible amount of work on 3-D navigation algorithms available in the open literature, the majority of the vertical tracking techniques of this research are unique and covered in Chapter III.

1.3 Goal

Using computer simulations or robotic implementations in the design of navigation algorithms pose their own unique difficulties. Developing a system for robotic implementation can be costly as well as extremely time consuming. Both of these

constraints were limitations in developing this thesis research. While developing an accurate and computationally feasible chemical plume is a limitation in the development of computer simulation based odor tracking navigation algorithm, the simulated plumes can be simplified making the computer simulation environment feasible. Thus, a better place to begin the development of a cutting edge architecture is using computer simulations, which is the direction taken in developing both the 2-D and 3-D navigation algorithms. A list of the objectives for this research are:

1. Develop a 2-D simulation environment equivalent to that of the wind tunnel experiments covered in [50]. This includes a dynamic 2-D plume model and dynamics model accurately depicting the capabilities of a moth.
2. Design a 2-D odor-based navigation algorithm constrained to the capabilities of a moth. Compare these results with those of the real moth wind tunnel experiments of [50].
3. Develop a 3-D simulation environment representing a small unmanned aerial vehicle (UAV) traveling a horizontal distance of up to 9,000 ft and a vertical distance of up to 3,000 ft. This includes a 3-D dynamic plume model and a dynamics model accurately depicting a small UAV.
4. Design a 3-D odor-based navigation algorithm constrained to the capabilities of a small UAV. Adjust design parameters to minimize flight time and maximize success rate.

1.4 Approach

1.4.1 Chapter II: Background. This chapter gives a comprehensive background of the research conducted on the moth's odor-based navigation capability. It also presents a thorough review of the most compelling attempts to develop a 2-D bio-inspired navigation algorithm. Both of these topics are important in understanding the reasoning behind the odor-based navigation algorithms developed in this thesis research.

1.4.2 Chapter III: Simulation Development. The design of both the 2-D and 3-D simulations and navigation algorithms are presented in detail. The 2-D case is discussed first, as it lays the foundation for the 3-D case. Both discussions are decomposed into 4 sections: plume model, dynamics model, sensor model, and navigation algorithm.

1.4.3 Chapter IV: Simulation Execution. The design and execution of Monte Carlo simulations for both the 2-D and 3-D cases are presented in Chapter IV. Appropriate variables are modified in an attempt to obtain the best results: replication of the moth wind-tunnel test for the 2-D case and the quickest, most accurate for the 3-D case.

1.4.4 Chapter V: Results and Conclusions. This chapter summarizes the results from the simulations discussed in Chapter IV. Conclusions are drawn from these results and recommendations made for any continuing research on this topic.

II. Background

There were three primary areas that need to be researched before any work on developing the navigation algorithms could begin:

1. Conduct a thorough review of any research related to the observations of the male moth's behavior when tracking pheromone plumes.
2. Develop a thorough enough understanding of fuzzy logic so as to have the capability of designing a fuzzy controller for use in the navigation algorithms.
3. Review any work related to the development of a robotic, odor-based navigation system or a computer simulation testing an odor-based navigation algorithm.

The results from this background literature review are discussed in the following sections of this chapter.

2.1 Biology

As discussed in Chapter I, many animals are able to locate food or a member of the opposite sex by tracking chemical plumes. The research discussed herein focuses on the ability of male moths to locate females by tracking the sex-attractant pheromone they release. Moths' antennae are equipped with thousands of small hairs giving it the ability to detect the pheromone up to 100 meters away [5, 49]. The Tobacco Hornworm moth (MSexta), as seen in figure 2.2, has been widely studied regarding its ability to locate a female through turbulent air flow via the female's pheromone plume. The structure of the moth's flight path remains consistent among different subjects, and the age of the moth does not ostensibly affect the outcome. The MSexta's life span, approximately seven days [41], does not allow it time to learn how to navigate a pheromone plume. This instinctual or innate behavior, apparently under primarily genetic control, allows the moth to successfully navigate a plume to its source on the first try. This reemphasizes that learning is unlikely to be an important factor in a moth's capability of navigating a pheromone plume [5, 49]. The quicker a male moth can locate the female, the better his chances to mate and pass on

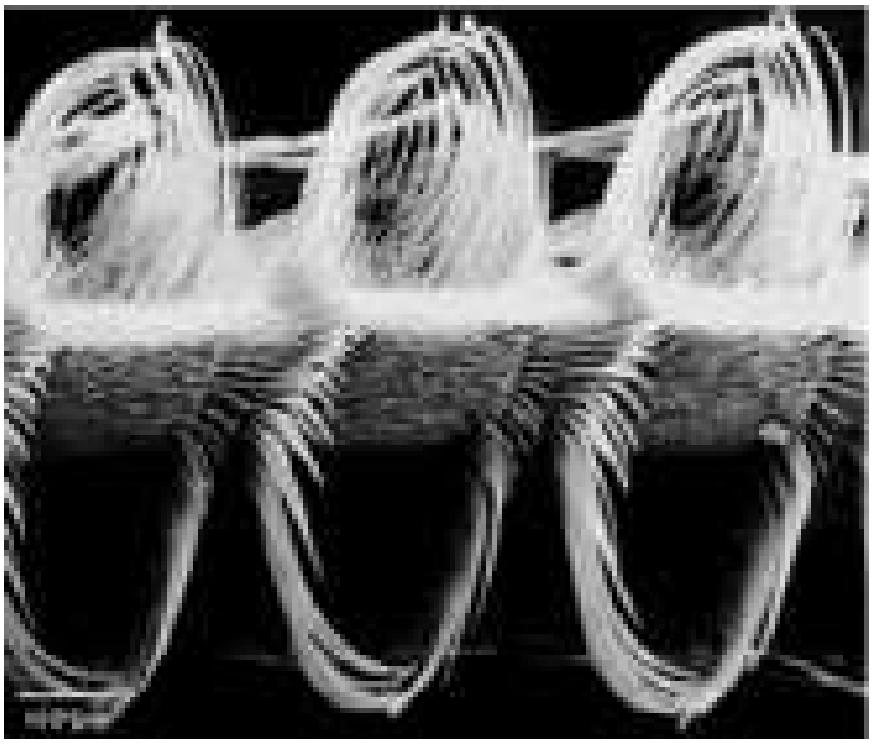


Figure 2.1: MSexta Antenna. [10] (This figure was replicated from [10]).

his genes. In the laboratory wind tunnel environment, roughly 70% of the time, the male reaches the source of pheromone [5]. This odor-based tracking phenomenon is a potential stepping-stone from which engineers can develop more advanced odor-based navigation algorithms.

The typical structure of a MSexta's flight profile while tracking a plume begins with the initial contact of pheromone. Upon such contact the male moth, who is most certainly downwind from the pheromone source, immediately maneuvers into the wind and begin an upwind movement. The moth uses visual flow fields to calculate its ground speed and heading with respect to wind direction. A common hypothesis among experts in this field is a moth's image flow consists of both longitudinal (current heading) and transverse (orthogonal to current heading) components [See Figure 2.3] [2, 9, 25, 32, 33]. This process is known as optomotor anemotaxis [18]. Figure 2.4 shows a depiction of a moth's flight profile exhibited during its upwind, pheromone tracking behavior. The moth's pheromone tracking behavior consists of

Figure 2.2: Three Stages of MSexta. The Larva (18-23 days), pupa (19-23 days), and adult (7 days) stages of MSexta are shown [41] (This figure was replicated from [41]).

3 main components: Casting; Counterturning; and Surging. Casting occurs when the insect has lost contact with the pheromone plume. It increases it's speed and flies perpendicular to the direction of the wind, increasing it's chances of once again encountering the pheromone plume. Counterturning is an "in the plume" behavior as the moth moves in a zigzag pattern while maintaining upwind progress. Surging is a more narrow version of Counterturning, occurring as the moth detects an increased concentration of pheromone. While making upwind progress, the moth cuts a zigzag pattern across the direction of the wind [18]. However, the degree to which the moth travels across the wind can vary significantly. If the moth loses contact with the pheromone, the counterturning behavior covers a larger crosswind area. This casting behavior results in larger crosswind movements with turns potentially increasing past 180°. Such behavior typically leads to zero upwind, or even downwind displacement, as the moth tries to relocate the plume. Casting continues until the moth either detects the pheromone again or it gives up [2]. As the moth detects higher concen-

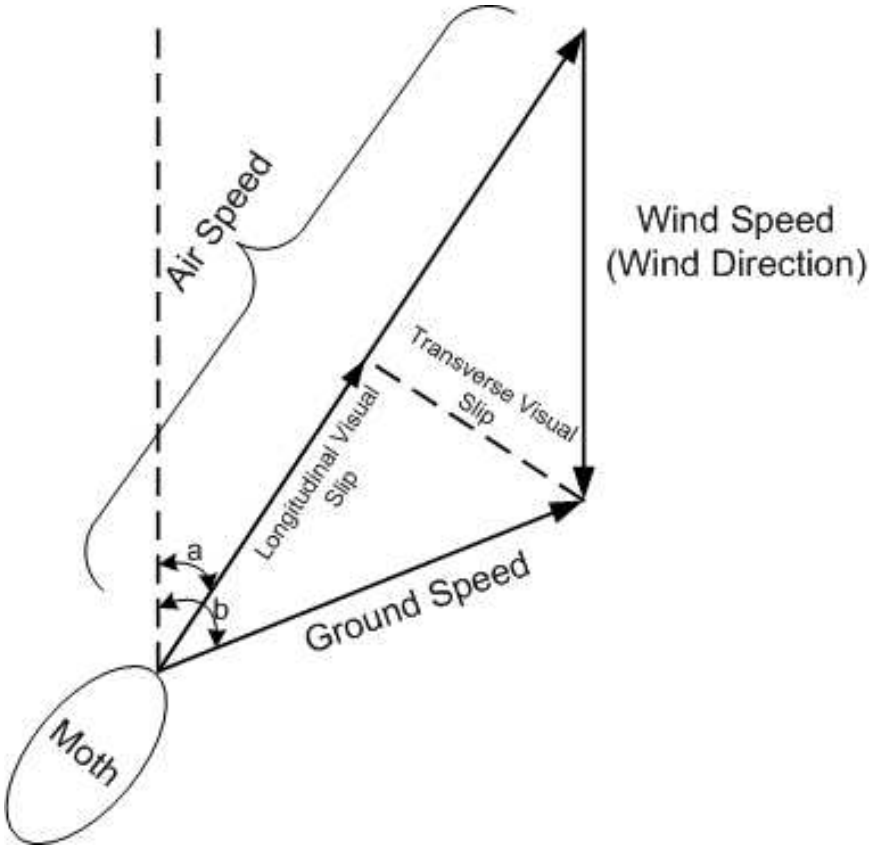


Figure 2.3: Optomotor Anemotaxis. When navigating a pheromone plume, moths appear to maintain both a set track angle (a) and ground velocity with respect to the wind direction and velocity. This is accomplished by the moth's ability to calculate the transverse and longitudinal components of visual slip using visual inputs [2, 9, 25, 32, 33] (This figure was reproduced from [2]).

trations of pheromone its crosswind movement decreases resulting in a surge to the source [2].

The characteristic features of the MSexta's flight profile while navigating a pheromone plume are not major topics of discussion among the fields' behavioral experts. Instead, the source of debate concerns the mechanisms underlying the control of the behavior, the sensory inputs that the animals use to control their behavior, and functional significance of the different components of the behavior (ie., why the counterturning behavior, why the crosswind component of the behavior, etc.) From

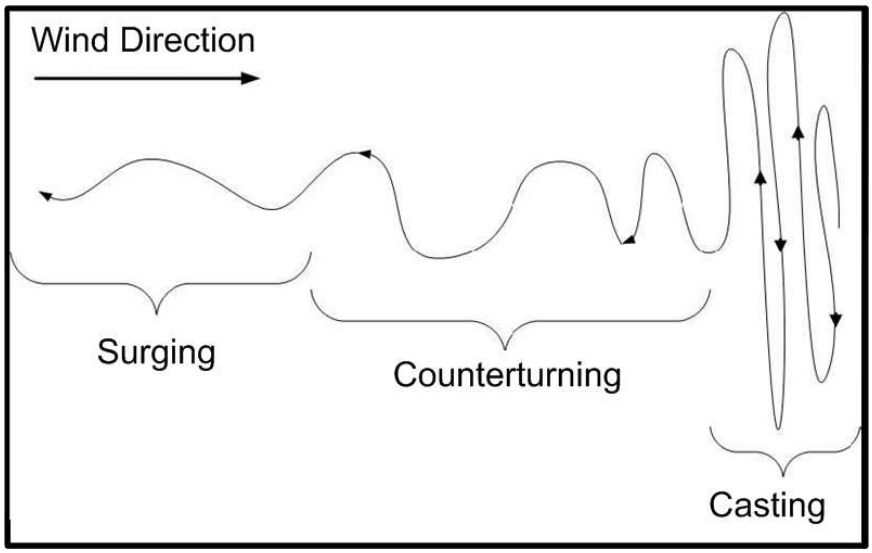


Figure 2.4: Example of a Moth's Pheromone Tracking Behavior. The moth's pheromone tracking behavior consists of 3 main segments: Casting; Counterturning; and Surging.

an engineering perspective, the hypotheses based on experimental data, preferably in a repeatable and controllable environment, are the most useful.

There have been many experiments confirming that the zigzag pattern, in the counterturning behavior, is an intended behavior. The wing muscles of moths have been monitored during flight with results indicating that the moth is trying to maintain the zigzag profile [2, 48]. These results weaken the Preiss and Kramer hypothesis [2,32,33] that moths are attempting to maintain a 0° heading (into the wind) with errors in detected wind speed and direction causing the zigzag flight profile.

One common hypothesis suggests moths have an internal mechanism which controls the amount of time between turns [3,19,20]. It has been repeatedly displayed through wind tunnel experiments that this regularity in timing between turns is a real phenomenon. Small wind tunnels (1 m $\times$ 2 m) have been used in conjunction with high-speed video cameras to further analyze the characteristics of the "in-plume" counterturning behavior. In MSexta, the average time between turns is on the order of 500 to 550 ms with the largest variations of 466 to 833 ms [50]. While this regularity occurs in many of the moth species studied, the actual timing varies [2].

In addition to the inter-turn interval the MSexta also appears to reduce its ground speed and increase the frequency at which it crosses the wind line when approaching the pheromone source [6, 49, 51, 52]. This behavior prompted additional wind tunnel experiments in an attempt to correlate characteristics of the counterturning behavior with different structures of pheromone plumes. The test assumed the concentration detected and frequencies of detections are important inputs in the moth's navigation scheme. Experiments showed that when increasing concentrations of the pheromone source, regardless of plume structure (benign versus turbulent), the moth would fly slower, more narrow profiles [6]. Also, when the source concentration was kept constant and the plume structure changed from less to more turbulent, the moth's ground speed would increase and its counterturns would become wider and less frequent [6, 23, 27, 52].

Wind tunnel experiments have shown ground speeds to fluctuate from 0 $\frac{\text{cm}}{\text{s}}$ to 115 $\frac{\text{cm}}{\text{s}}$ regardless of wind speed. More than 80% of the moths studied reached peak ground speeds at the midpoint between counterturns [50]. The results from this specific study strengthen the hypothesis that moths react, changing their flight characteristics, in response to the detection frequency of pheromone. In addition, this experiment suggests the concentration of pheromone detected is an important factor in a moth's ability to navigate a pheromone plume [6].

To further emphasize the importance of time between detections, studies have been done which measure the pheromone detection rate while the moth is in flight. Detection rates above 5 Hz typically promote upwind flight with increased velocity, while rates of pheromone detection below 5 Hz dictate a slower upwind velocity [21, 27, 44]. Some species of male moths are able to distinguish up to 10 $\frac{\text{detections}}{\text{s}}$ [21, 38] and have reaction times in response to detecting pheromone within 200 ms [21, 26, 27].

Said experiments and associated results allow only a glimpse into the complete understanding of a moth's odor-based navigation abilities. From this point forward it is up to the engineers and biologists to hypothesize how a moth uses the information

it senses in order to determine its next course of action. From a biology perspective, it is important for engineers attempting to model an animal's behavior to keep in mind that just because information is available to the animal does not mean the animal uses it [6]. In fact a common principle of biological sensory systems is that they actually filter out information that is not necessary to support the survival of the organism (discussions with Dr. M.A. Willis). However, when trying to engineer a system inspired by nature, we are only limited by technology versus the limits of the creature.

2.2 Fuzzy Logic

There are many uncertainties an engineer has to deal with when designing a control system that is supposed to mimic a natural phenomenon. Typically, there is very little information one can gather on the exact details of a natural phenomenon such as how a male moth successfully navigates a pheromone plume. As discussed in Section 2.1 of this chapter, experiments have been conducted in an attempt to generate logical hypotheses on how a moth navigates. Muscles can be monitored to determine if a particular mode of flight is being stabilized or environmental variables can be controlled to elicit responses to certain environmental conditions. However, the ability to precisely monitor every part or function of a moth would be a daunting task and has yet to be accomplished. This leaves much room for biologists and engineers to hypothesize how the moth is exactly processing his inputs in order to make the decisions necessary to navigate the pheromone plume. These ambiguities give merit to the idea of using a fuzzy logic based controller to accomplish the task of navigating a simulated aircraft through a chemical plume to the location of its source [35].

An aspect of fuzzy logic which makes it a desirable engineering tool in designing control systems is the use of logic rather than equations in describing the system to be controlled. It would be extremely difficult, if not impossible, to develop an accurate set of mathematical equations from which one could precisely model a moth's odor-based navigation behavior. This ambiguity in how the real system operates is more easily

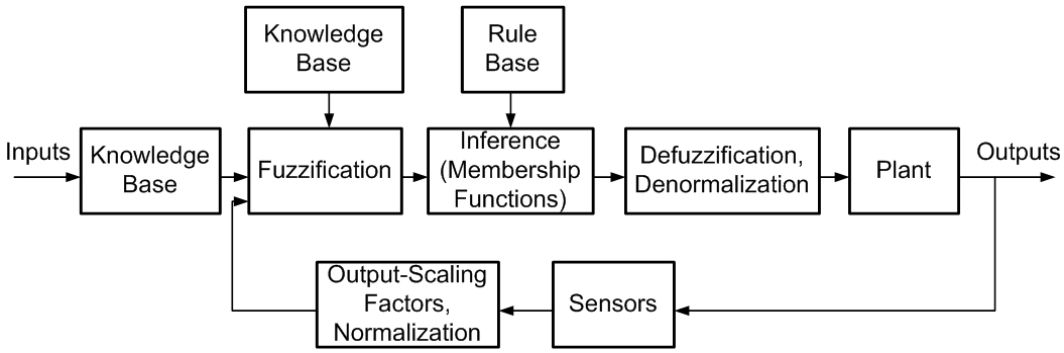


Figure 2.5: Non-Adaptive Fuzzy Controller Flowchart (This figure was reproduced from [35]).

built into the fuzzy logic. However, using fuzzy logic implies the control system is not optimal, but should have the ability to be tuned properly to generate an acceptable level of performance.

Figure 2.5 is an illustration of a control system which utilizes a non-adaptive fuzzy controller in order to send control inputs into a desired plant [35]. One's knowledge of how a system functions and the critical inputs driving the system, dictate the number of input fuzzy sets needed and how their associated membership functions are designed. Inputs into a fuzzy controller must be normalized to meet the parameter bounds of the input fuzzy sets. The knowledge of how a system uses certain inputs dictates the how the normalized inputs are fuzzified. This "fuzzification" is accomplished by the input fuzzy sets and corresponding membership functions. The understanding or inference of how a system combines its inputs to generate specific outputs is reflected in the design of the fuzzy rules. The outputs of a fuzzy controller must then be denormalized, or defuzzified, to meet the input requirements of the plant to be controlled. Therefore, the design of a fuzzy controller relies on the use of three main tools: fuzzy sets, membership functions and fuzzy rules.

Fuzzy logic is just one approach in solving the odor-based navigation problem. Other potential methods, not taken in this thesis work, are genetic algorithms, neural networks, adaptive neural networks as well as combinations of these techniques.

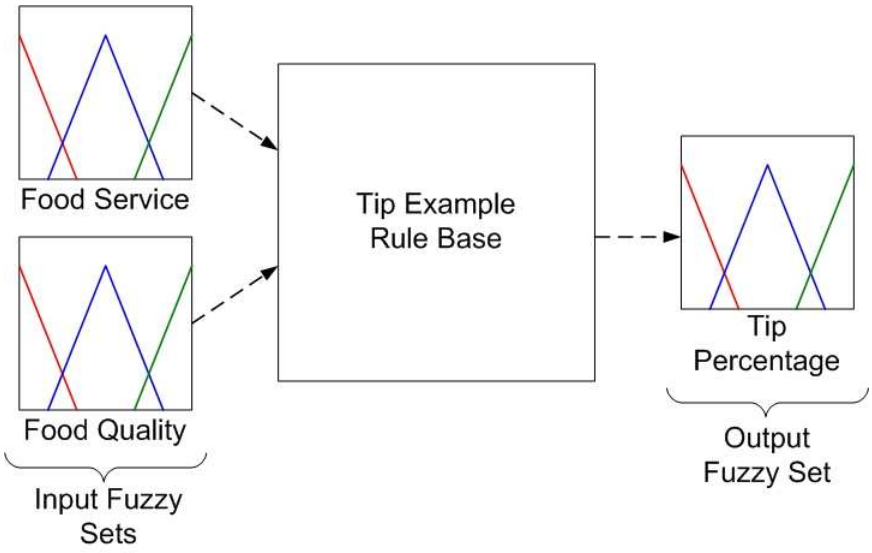


Figure 2.6: Tip percentage fuzzy controller. Two inputs drive the tip percentage system "Food Service" and "Food Quality." The rule base uses the fuzzified inputs to generate a given output.

2.2.1 Fuzzy Sets. Fuzzy sets correspond to both input and output parameters [1, 14, 35, 36]. For example, if a fuzzy controller was being designed to determine the percentage to tip a waiter or waitress, the input fuzzy sets may be service quality and food quality [1]. The output fuzzy set of the same example would be tip percentage. Figure 2.6 shows a simplified layout of a fuzzy controller designed to generate a tip percentage given the two input fuzzy sets "Food Service" and "Food Quality."

2.2.2 Membership Functions. Membership functions give a means to describe how the data contained within a fuzzy set is categorized. Using the tip generation example, the "Food Service" fuzzy set may contain 3 membership functions describing the service quality as poor, average, or excellent. In order to make sure the inputs fall within the specified limits of the fuzzy set, they must be bounded or normalized to fit within the minimum and maximum values that define the set's space. In short, every input into a fuzzy set must fit within their set numerical range. The determination of whether service was poor, average, or excellent is not a black and white decision, it is fuzzy. Figure 2.8 is an example of how the membership functions

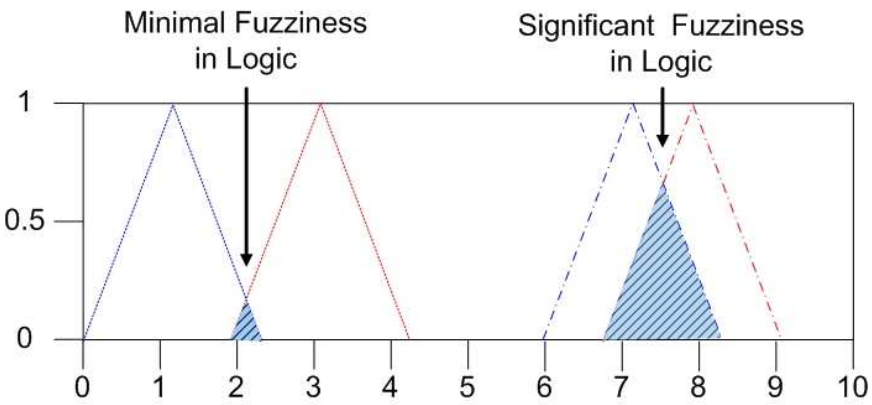


Figure 2.7: Range of fuzziness. In designing a fuzzy controller, engineers have the ability to make clear decisions (minimal fuzziness) or clouded decisions (significant fuzziness) (This figure was reproduced from [1]).

for the "Food Service" fuzzy set may be designed. The overlap between the functions is a description of the fuzziness in the decisions to be made within that particular fuzzy set. The more overlap amongst membership functions indicates a high degree of fuzziness in the decisions being made. Figure 2.7 illustrates the levels of fuzziness which can be incorporate into a fuzzy controller.

The membership functions given in the example illustrated are simple triangular functions. More complex functions can be used to describe the fuzzy sets [1, 14, 35, 36]. Some of these functions are: bell curves, Gaussian distributions, and sigmoid functions. All of which are depicted in Figure 2.9.

The output fuzzy sets and associated membership functions are generated in the same manner as the inputs, except the outputs must be bounded to meet the requirements of the controller they are meant to control. In the tip example, the output fuzzy set is bounded in order to set a maximum and minimum tip percentage given certain circumstances. Figure 2.10 shows an example of the output fuzzy set with its associated membership functions. Again, the overlap between membership functions shows the decision of giving a high tip versus an average tip is fuzzy.

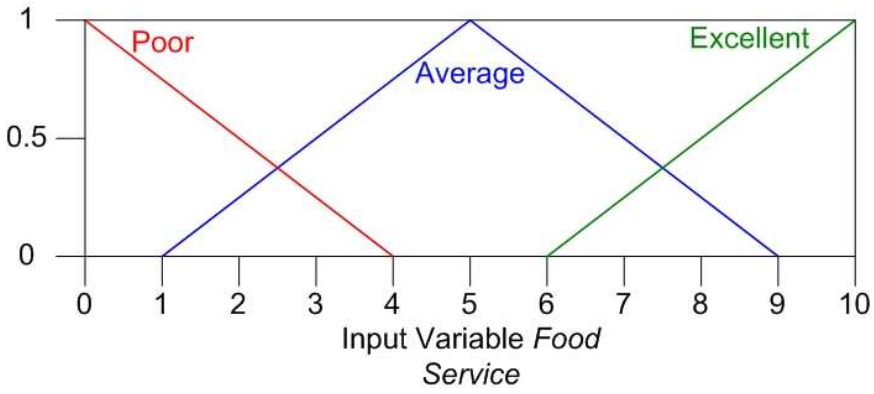


Figure 2.8: Food Service fuzzy set. Simple triangle functions were used to describe the range associated with the level of service quality. The inputs to this fuzzy set must be normalized to a value of 10, the maximum value allowed. There is significant overlap between the membership functions describing the fuzziness associated with determining the quality of the "Food Service" input.

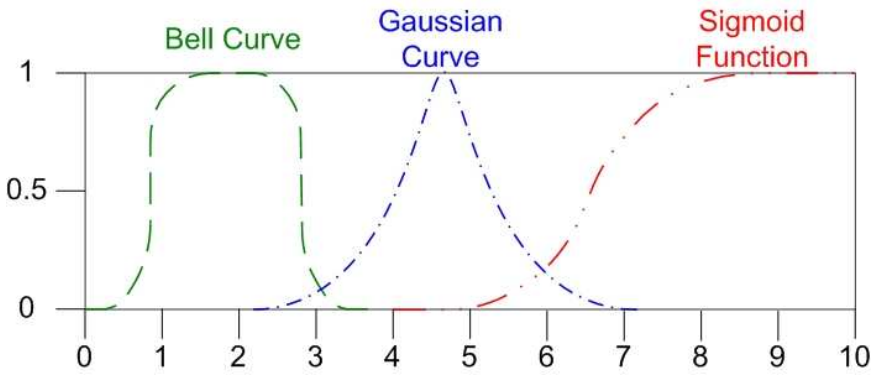


Figure 2.9: Examples of membership functions. Membership functions can be designed using a number of different functions in addition to triangular. Three commonly used functions are the bell, Gaussian, and sigmoid (This figure was reproduced from [1]).

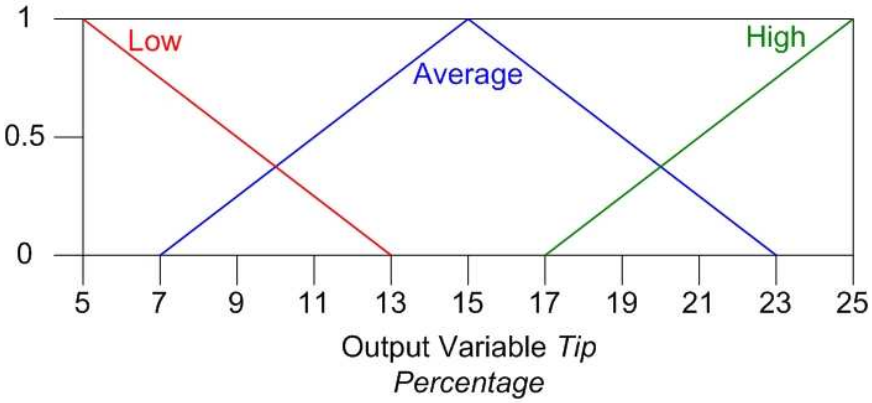


Figure 2.10: Output fuzzy set, Tip Percentage. Simple triangular functions were used to divide the Tip Percentage fuzzy set into three categories: Low, Average, and High. The percentage of a tip can range from 5% to 25%, depending on the quality of the food and service.

2.2.3 Fuzzy Rules. A set of rules need to be generated in order to map inputs to outputs. The rules are generated by using easy to understand "If-Then" statements. Although not complex this structure allows for a wide range of control when mapping inputs to outputs. A rule base associated with the tip example is shown in Figure 2.11. In this example, the aggregation operator AND is used throughout. However, the OR operator could also be used. Each input membership function must be represented in the rule base. In addition, all permutations of fuzzy inputs and associated membership functions must be accounted for in the rule base, also known as antecedents. All output fuzzy sets and associated membership functions should be represented at least once, also known as consequents. If this is not the case, a potential output which is never used was included, wasting time and effort during the design process.

There are multiple inference methods for generating the defuzzified output [1, 14, 35, 36]. This research uses the Mamdani method, chosen due to ease of implementation. The MATLAB® fuzzy toolbox suggests using the Mamdani method and makes it the default inference method. The Mamdani method consists of the following steps:

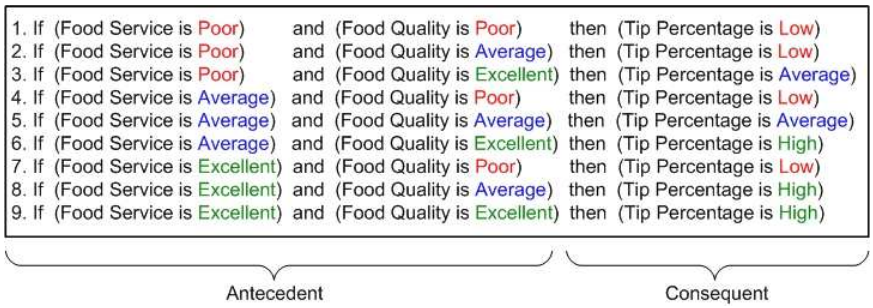


Figure 2.11: Fuzzy Rules.

1. After fuzzification of the inputs occurs (Figure 2.12), execute all fuzzy rules in parallel (see Figure 2.11 for example fuzzy rules). Using the AND aggregation operator results in taking the minimum value of any membership function associated with the same rule.

$$Min(0.2, 0.4, 0.6) = 0.2 \tag{2.1}$$

Using the OR aggregation operator would result in taking the maximum value of the associated membership functions rather than the minimum.

$$Max(0.2, 0.4, 0.6) = 0.6 \tag{2.2}$$

2. The value determined by Step 1 is the weight to be applied to the appropriate output fuzzy set and associated membership function(s). This process is known as implication and is illustrated, along with Step 1, in Figure 2.13.
3. The third step, aggregation, consists of comparing consequents of all the rules and taking the maximum value at each point across the entire range of the output fuzzy set. Assuming that there are only three rules associated with the tip example, Figure 2.14 shows how this step works.
4. The final step deals with the defuzzification of Step 3. This research uses the centroid method, finding the center of area under the curve, resulting in a single, defuzzified output value. This process is illustrated in Figure 2.14.

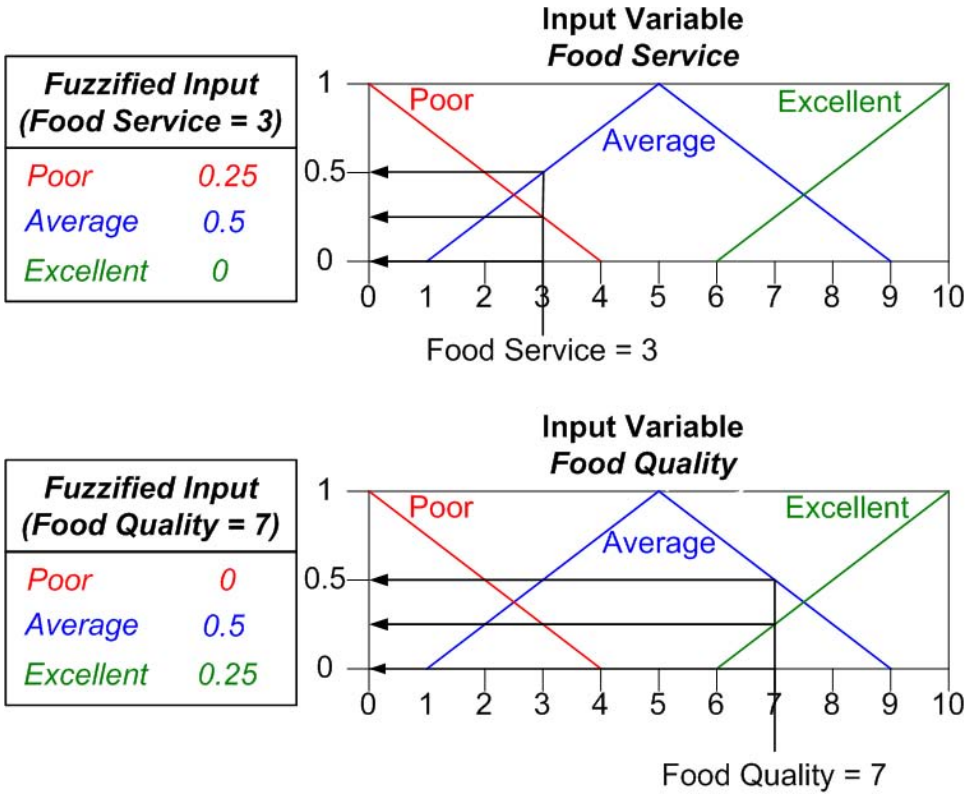


Figure 2.12: Fuzzification of inputs. Given a "Food Service" input = 3 and a "Food Quality" input = 7, all membership functions within each fuzzy set have an associated fuzzy value (This figure was reproduced from [1]).

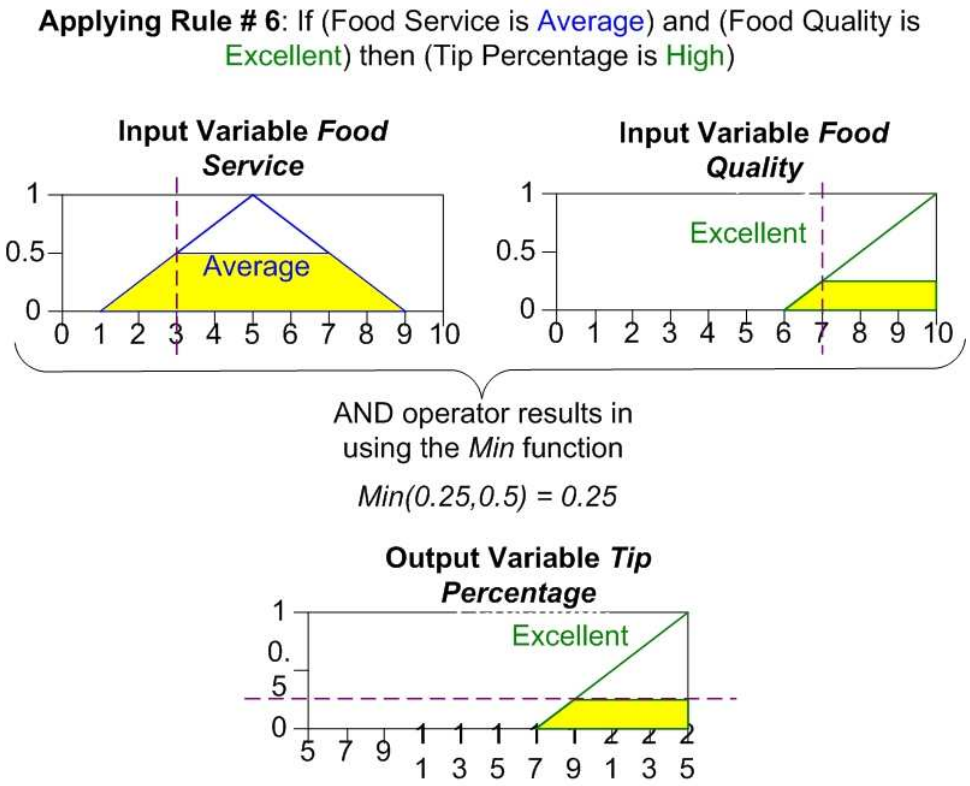


Figure 2.13: Implication process of a fuzzy controller. Given "Food Service" input = 3 and a "Food Quality" input = 7 the associated "Average" membership function and "High" membership function have fuzzy input values of 0.5 and 0.25 respectively. Rule 6 uses the AND aggregation operator resulting in an evaluation of the fuzzy inputs with the *Min* function. The resulting value of 0.25 is now the weight applied to rule 6's associated fuzzy output membership function, "High" (This figure was reproduced from [1]).

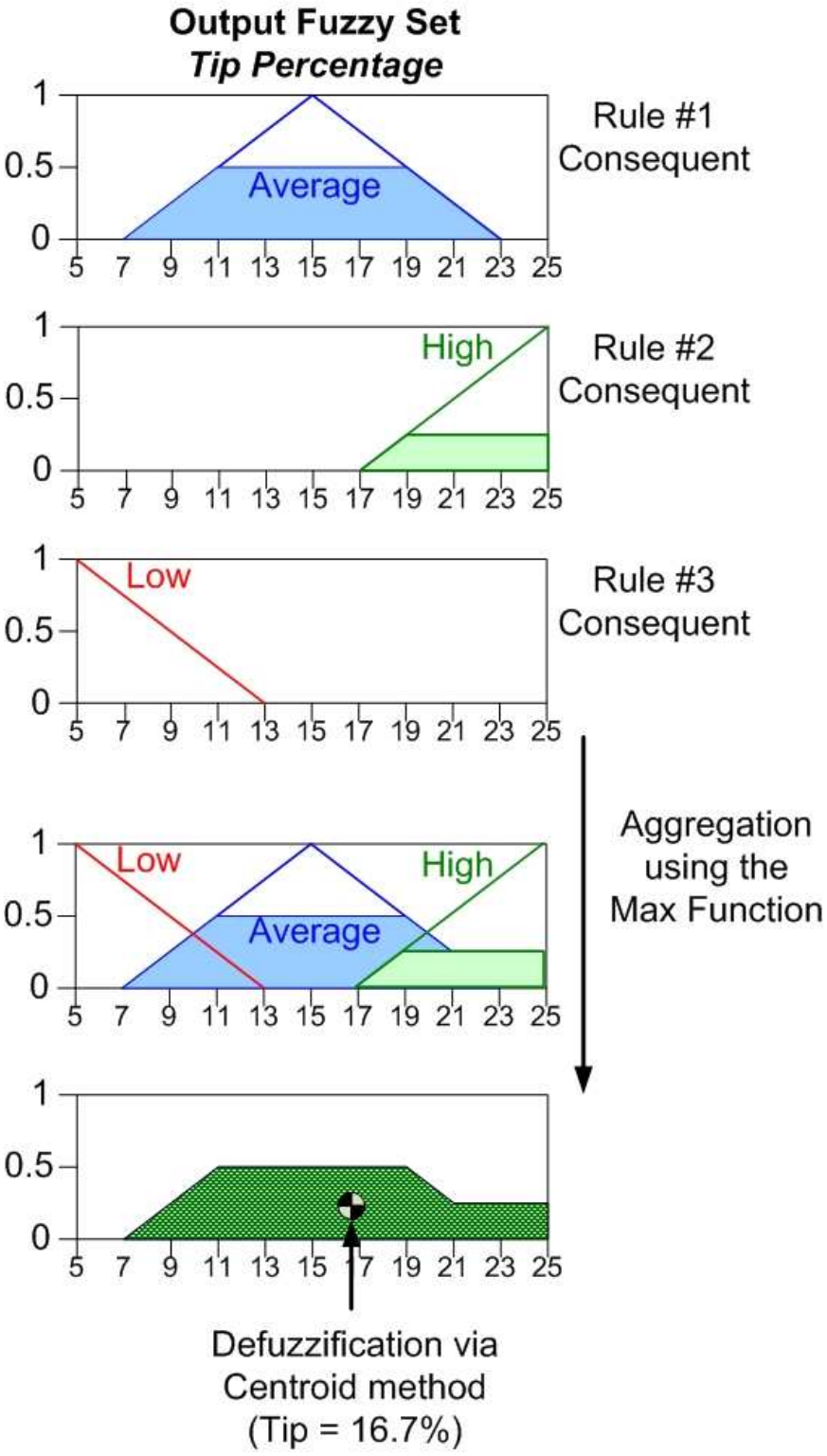


Figure 2.14: Aggregation and defuzzification. Taking the maximum values of 3 consequents results in a curve corresponding to the fuzzy, suggested tip percentage. One can determine the defuzzified tip percentage by finding the centroid of the area under the curve (This figure was reproduced from [1]).

2.3 Simulations and Robotic Implementations

Over the last 20 years, there have been several attempts to develop either computer simulations or robotic systems with the purpose of autonomously navigating chemical plumes in order to locate their source [5, 12, 16, 21, 29, 43]. The majority of these designs efforts have been focused on robotic implementations. The difficulty in generating realistic, time varying plume structures is the most compelling reason why researchers have spent limited time and resources in developing such computer based simulations. The problems of plume modelling stem from the inability to accurately model turbulent airflow and the associated turbulent diffusion of the chemical [7]. By using a variety of chemicals, associated sensors, and robotics platforms, researchers have been able to design experiments in controlled environments as well as uncontrolled, or natural, environments to test odor-based navigation schemes [12, 16, 17, 24, 29].

The use of robotics and sensors allows true chemical sources to be placed in turbulent wind flow, generating a real turbulent chemical plume. As such, these "real world" experiments tend to have more validity than computer simulations [21]. However, the generation of computationally feasible simulations, incorporating chemical plumes with properties similar to those of real plumes, is an effective way to test potential algorithms before spending the resources necessary to build robotics systems that may be destined to fail. Incorporation of a true-to-life dynamics model of the intended robotics platform adds additional validity to the simulation. For example, one would not want to use a dynamics model of an aircraft which allows maneuvers that a real UAV could not perform. If this occurred, the navigation scheme developed would likely lack validity for a real world application. The following subsections contain a smattering of different simulations and robotic implementations designed to solve the odor-based navigation problem.

2.3.1 Li Implementation. Li et al. [21] have executed a comprehensive two-dimensional simulation study. Their odor-navigation techniques and strategies were

inspired by the behaviors of male moths tracking a pheromone plume. However, they did not restrict their simulation to a moth's capabilities as they did not set out to mimic its behavior. The foundation to their simulation, as should be with any odor based simulation, was the dynamic plume model they incorporated. The plume model took three main structural attributes into account: the concentration detected over time at a fixed position should increase and decrease in strength in accordance with experimental data; the shape of the plume should be sinusoidal and vary with time; and the shape of the plume and airflow should not posses any jump discontinuities. This simulation released a sequence of puffs into a wind stream with each puff being composed of a given number of "pheromone" filaments. A more detailed description of this plume model can be found in Farrell et al.'s work [11]. For their Monte Carlo analysis (executing multiple runs of a simulation while randomizing certain variables), they used two types of plumes: narrow and wide.

Li's navigation scheme consisted of four components: locating the plume; maintaining contact with the plume; reacquisition of the plume; and "declaring" the source found. The components of most interest are maintaining contact with the plume and reacquiring the plume. The simulated sensor functioned as a binary detector. In other words, the concentration detected played no role in the navigation algorithm. The simulation was constrained to a 100 m $\times$ 100 m square. The methodology for maintaining contact with the plume uses the following variables in its decision-making process:

1. T_{lost} = time when declared plume lost
2. T_{last} = time of last detection
3. T_{first} = time of first detection
4. β = heading relative to wind direction, taken after T_{first}
5. γ = heading relative to wind direction, taken after T_{lost}

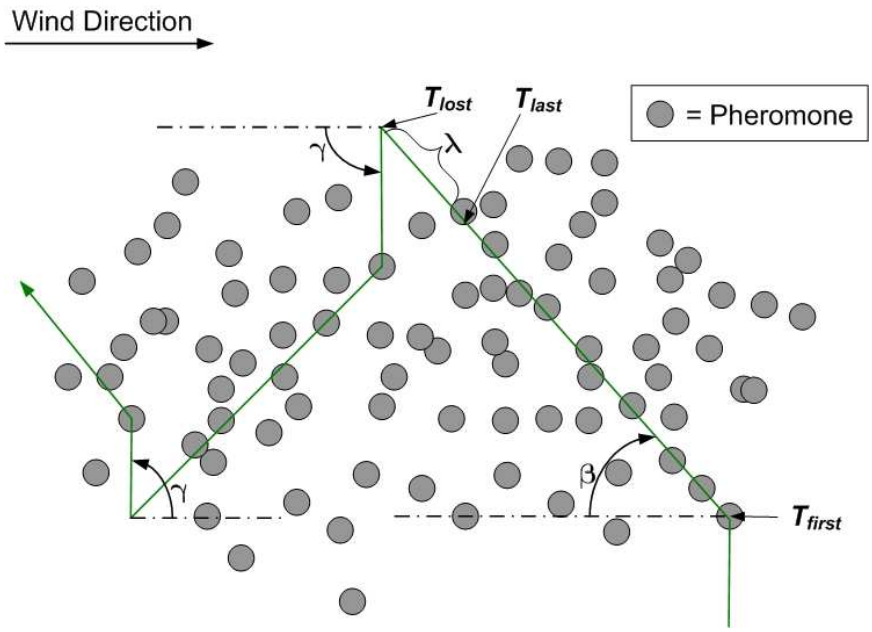


Figure 2.15: In-plume tracking behavior. If a certain amount of time, λ, transpires without a pheromone detection the vehicle will fly perpendicular to the wind line until it once again detects the pheromone. When no pheromone is detected after a duration of λ sec, the vehicle will move β degrees from the wind line, making upwind progress. This path will continue until a detection is not made within λ s (This figure was reproduced from [21]).

6. λ = threshold of time after T_{last} to declare T_{lost}

$$\lambda = T_{lost} - T_{last} \tag{2.3}$$

Figure 2.15 is an illustration of what a track may look like, incorporating the variables mentioned.

The researchers varied λ to observe the resulting affects on the amount of time spent within the plume. These variations have both positive and negative effects on the trajectory. A larger λ increases the chance that a detection will occur. However, this also allows for a greater distance travelled outside of the plume's boundaries.

Owing to the unpredictability of the plume structure, it is inevitable that the search vehicle at some point loses track and needs to reacquire the plume. The authors use time, T_w, as a boundary condition to signify that the vehicle has left the plume

and needs to transition to a reacquisition navigation scheme. The authors found that searching perpendicular to the wind gave the quickest plume recovery time. This is reminiscent of a moth's casting behavior as previously discussed. Simulations were run to find how β affected plume maintain time, T_m. The results concluded that while $\beta > 10^\circ$ provided satisfactory performance, a time varying β performed better in maintaining contact with the plume. This does propose a trade-off between upwind movements and T_m. The most successful strategy developed is depicted in Figure 2.15, incorporating a time varying β.

The simulation consisted of a 100 m $\times$ 100 m area of operation (AO). The UAV was placed 40 m downwind from the source and moved at an average speed of 1 $\frac{\text{m}}{\text{s}}$. The results of this study were successful, 97% of UAV's succeeded in locating the source within a 1 m radius and under a 300 s time limit. This technique was also robust enough to successfully navigate simulated plumes of variable densities as well as true water flume data.

A couple of issues in the presentation of this research include the lack of discussion of the UAV dynamics model used in the simulations and a less than thorough discussion of how the endgame success was measured. UAV dynamics will affect which maneuvers are able to be made, thus affecting values for β, λ, T_w, etc. The authors mention the ability of declaring the location of the source, however, the bulk of the research suggests merely travelling within a 1 m radius of the source is a declaration of success. Given such limitations, the most successful algorithm developed by the authors is a good starting point for an actual two-dimensional robotic implementation.

2.3.2 Marques Implementation. Marques et al. [29] studied the performance of three two-dimensional, bio-inspired navigation strategies implemented on an autonomous, land-based, mobile robot. The robotic platform consisted of a gas sensor employing an electronic nose and wind sensor. The electronic nose is made up of a chemical sensor and pattern recognition software used to detect the correct chemical compounds in the sensed air.

2.3.2.1 Biased Random Walk Algorithm. The first algorithm uses the biased random walking strategy of bacteria. Bacteria move in straight lines followed by short movements in the opposite direction. These short periods of reversed directional motion cause a natural randomization of the direction of the next straight line motion [13, 29]. This leads to a simple algorithm easily expressed in the following pseudo code, where m is a distance defined by the user:

```
If  (current concentration > last concentration)
    Turn (+/- Random(5 deg)
    Move Forward (m +/- Random 5 percent of m)
Else
    Turn (+/- Random 180 deg)
    Move Forward (random 5 percent of m)
End
```

2.3.2.2 Silkworm Moth Algorithm. The second algorithm tested was developed based on the Silkworm moth's behavior associated with tracking a pheromone plume, similar to that of MSexta. The key behaviors include:

1. Upon contact with pheromone, orient into the wind.
2. Conduct a zigzag pattern across the wind-line while maintaining upwind movement.
3. Loss of pheromone results in the execution of a circular maneuver back to where the plume was last detected.

A flowchart of how the Silkworm moth algorithm works is illustrated in Figure 2.16.

2.3.2.3 Gradient Algorithm. The third algorithm tested was a gradient-following technique (which was not bio-inspired). This navigation routine was developed to minimize the distance travelled to the source of the plume. A Gaussian plume

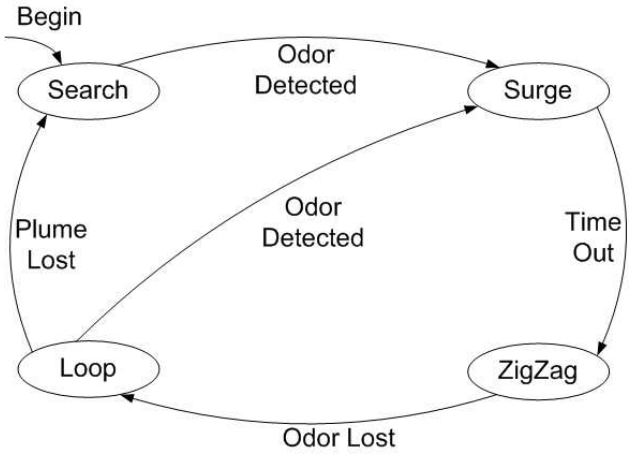


Figure 2.16: Marques et al.'s moth based algorithm (This figure was reproduced from [29]).

model was fit to the data taken by the robot and used to estimate the location of the source. This gradient-based algorithm is described in the following pseudo code:

```
Search for plume traces
While (odor detected)
    Estimate plume geometry
    If (concentration > threshold)
        Follow gradient
    Else
        Search for plume traces
    End
End
```

The experiment setup is illustrated in Figure 2.17 and included 20 iterations of each algorithm. The results shown in Table 2.1 illustrate that the Gradient and Silkworm moth algorithms were roughly equivalent in their outcomes. The Bacteria based algorithm was also successful in reaching the source but took much longer to do so.

Aspects of the results that may not be applicable to a larger or more turbulent environment are those associated with the gradient technique. The more turbulent

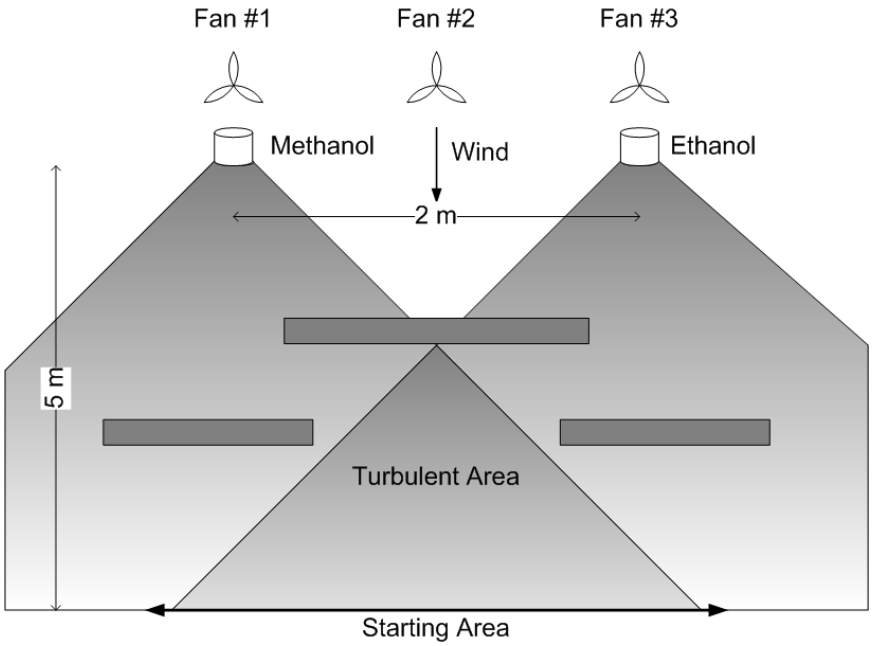


Figure 2.17: Marques et al.'s experiment set up (This figure was reproduced from [29]).

Table 2.1: Marques et al.'s experimental results. All the algorithms appear to have worked well. However, the experimental setup was fairly simplistic, minimizing the potential usefulness of the designed algorithms.

	Probability of Success	Average Time of Successful Run
Bacteria	90	243
Moth	100	89
Gradient	100	73

the atmosphere or when further from the source, a gradient technique would probably not perform as well due to the plume becoming less Gaussian in shape.

2.3.3 Vabø Implementation. Vabø et al. [43] developed multiple two-dimensional strategies designed to search for both an odor plume and the plume's source based on studies of Cod (Gadus morhua L.). The plume-searching strategy's objective is to locate the plume given no previous plume interaction has occurred. The source-searching strategies seek to navigate a detected plume to its source. These navigation algorithms were tested using a simulation of an underwater environment. In other words, propagating and navigating an odor plume through water instead of air. The time to find the plume was the measure of success for the plume-searching algorithm. Declaration of success for the source-searching algorithm was locating the source to within a 3 m radius along with the time it took to do so. The 3 m radius was based on the visual abilities of Cod.

The different algorithms employed for the plume-search were Counter Current (similar to casting of MSexta), Random Turn, Random Walk, and Upstream. The Random Turn method consisted of turning a random +/- N° every time interval, while the Random Walk method consisted of selecting a random direction to move for a specified amount of time. The Upstream method is self-explanatory.

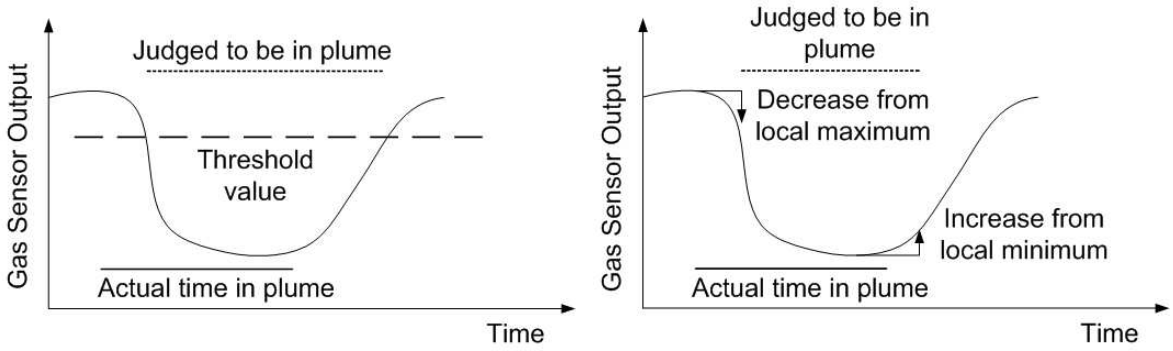
The navigation methods tested in the source-search were Upstream, Counter Current, and Gradient Search. Gradient Search works by moving in a straight direction until the detected concentration drops below a pre-defined threshold parameter. At this point, the vehicle turns randomly in a new direction and starts the process over. The Counter Current method is similar to the one above except the angles across the current line are arbitrarily smaller. A pseudo code representation of these navigation techniques would be ideal, however, [43] did not provide enough information to accurately portray the algorithms.

The most effective navigation method of finding the location of the odor source was the Upstream algorithm, closely followed by the Counter Current algorithm.

The plume-search simulations were tested using plumes of varying widths while the source-search simulations were only tested using a single type of plume (narrow and non-meandering). The successful results of the Upstream algorithm are likely due to the environment of the simulation. This algorithm would not be robust enough to handle wider or meandering plumes. However, if one knows they are close to the source, then the upstream algorithm may be a good tactic.

2.3.4 Ishida Implementation. Ishida et al. [16] developed a series of ground-based robots capable of navigating a chemical plume to its source. The capabilities and behaviors associated with moths navigating pheromone plumes were critical inspirations behind the navigation techniques developed by the authors. They changed from using a binary detection scheme to one that uses the transient response of the gas sensors. The binary detection method works. However, due to the sensors having a slow recovery time after making a detection, the robot is forced to move slowly in order to process the sensor information while maintaining contact with the plume. Using the transient response of the sensors allows for quicker response times and presents more information about where in the plume the robot is located (entering the plume, in the plume, exiting the plume). Figure 2.18 shows examples of the binary algorithm and transient response algorithm. The entire navigation algorithm is illustrated by the flowchart in Figure 2.19.

The initial search for the plume consists of the robot moving in a straight line. In an uncontrolled environment this would not be a successful search routine. However, the manner in which this experiment was designed guarantees this maneuver will lead the robot to the plume. Once the plume is detected, the robot moves to Phase Two, Upwind Tracking. During Phase Two, the robot moves upwind in a zigzag pattern dependent on the left and right gas sensor detections, turning towards the side with the higher concentration reading. The robot keeps track of the amount a sensor recovers from a detection. Should it fall below 10% from the last maximum detection, then Phase Four, local spiral search, is implemented. A backward spiral is executed



(a) Binary Detection Scheme

(b) Detection Based on Transient Response of Detector

Figure 2.18: Ishida et al.'s detection schemes for given chemical sensor's capabilities. (a) Binary detection scheme based on a set threshold for detection. (b) Using the transient response of chemical sensor to determine where the robot is located in the plume (This figure was reproduced from [16]).

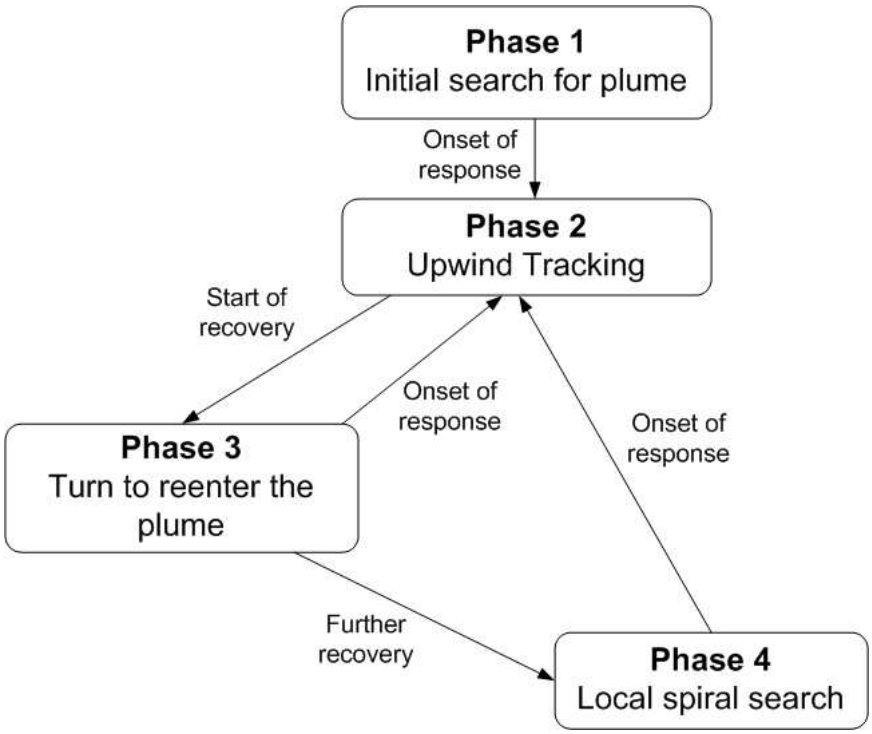


Figure 2.19: Flow chart of Ishida et al.'s odor based navigation algorithm (This figure was reproduced from [16]).

until the sensor detects an appropriate level of concentration. If nothing is detected, the spiral's radius increases, allowing the robot to cover a larger area in an attempt to reacquire the plume.

The technique discussed above is only applicable to this robot or those designed with similar sensors. However, the basic idea behind the tracking algorithm is to create an upwind zigzag pattern, a common idea presented frequently in the open literature [5,12,21,29,43]. Offshoots of the spiral routine have been used by a number of odor-based navigation schemes [12,21]. Although the method of using the transient response of the sensors is unique to the types of sensors used in this research, useful data may be gleaned from the event. It appears that considering the transient response instead of a binary on/off switch, provides a smoothed time history of finite, high frequency detections. This might prove useful for other sensors or simulations.

2.3.5 Farrell Implementation. Farrell et al. [12] have developed the most advanced and thorough robotic implementation of an odor based navigation system. The primary inspirations behind the algorithms used were from the behaviors of both moths and Antarctic procellariiform seabirds [12,30]. The authors' goals were to navigate and locate the source of an underwater chemical plume (Rhodamine dye) using an autonomous under water vehicle ((AUV)) located in a turbulent, near-shore, ocean environment. The chemical sensor used was strictly binary with a 10 Hz sampling rate. The scenarios implemented constrained the AUV to an area of operation of 367 m $\times$ 1094 m. The AUV's commanded speed was set at 2 $\frac{\text{m}}{\text{s}}$ with a fixed altitude of 2 m (2 dimensional experiment). A flowchart of the navigation algorithm is shown in figure 2.20.

The mission starts with the Go-To behavior, maneuvering the AUV to a desired starting location. The only information on the location of the source is that it resides within the operational area. The AUV moves across the current to one of the area of operation corners furthest down current. Once the Go-To command is executed, the Find Algorithm maneuvers from one edge of the operational area to the other. The

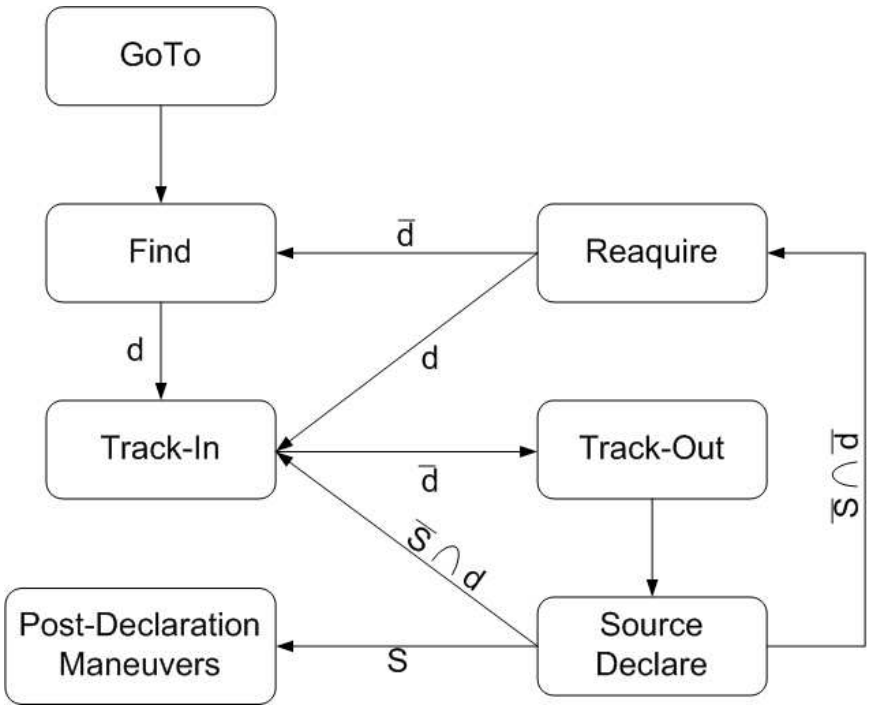


Figure 2.20: Flow chart of Farrell et al.'s odor based navigation algorithm. d = detection, $\bar{d}$ = no detection, S = source declared, and $\bar{S}$ = source not declared (This figure was reproduced from [12]).

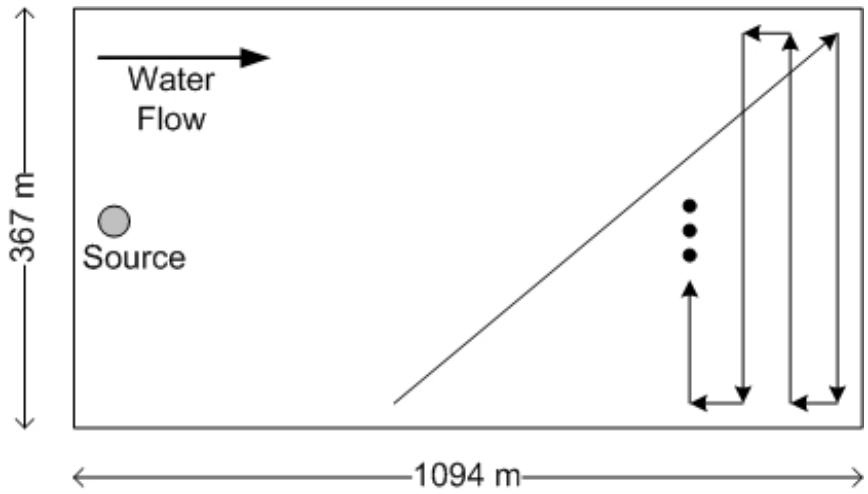


Figure 2.21: Illustration of initial search behavior incorporated into Farrell et al.'s odor based navigation algorithm. No matter where the AUV starts, it moves across the current to the furthest down current point of the area of operation. The AUV then maneuvers perpendicular to the current while making slight up current movement. This gives the AUV ample opportunities to detect the odor plume (This figure was reproduced from [12]).

bulk of the travel is done perpendicular to the current flow. This search method is indicative of a moth's search behavior. Figure 2.21 illustrates both the Go-To and Find algorithms.

Once a detection occurs, the TrackIn behavior is activated. The AUV navigates into the current plus/minus a set angle (β) from the wind-line. The authors used 20° for β. The direction of the robots trajectory changes every time a detection occurs. This behavior continues until a preset time (λ) passes in which no detections occurred. Every time a detection occurs, the robot saves the location where it occurred.

The TrackOut algorithm is executed when TrackIn times out. At this point the vehicle maneuvers across the plume to a set distance from the last detection which is furthest up-current. After completing this maneuver, if no detections occur, the Reacquire subroutine will be executed. However, if a detection does occur, the AUV either implements the TrackIn or Post-Declaration subroutines. The Post-Declaration behavior is invoked when the vehicle encircles roughly the same area more than once. This transition relies on the assumption that the AUV has encircled the source, cycli-

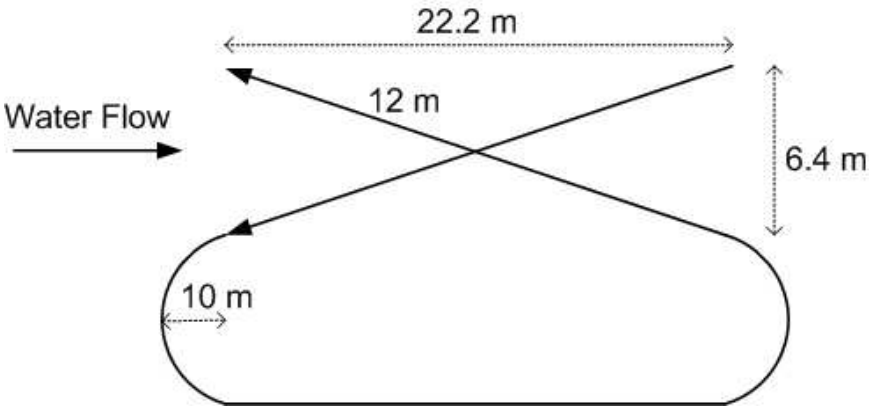


Figure 2.22: "Bow tie" behavior incorporated into Farrell et al.'s odor based navigation algorithm. This maneuver will attempt to reacquire a lost plume. This maneuver is centered 10 m up-current from the most up-current point on the list of detections. The AUV will execute as many "bow ties" as there were detections, giving it ample opportunity to reacquire the plume (This figure was reproduced from [12]).

cally coming in and out of the plume. At this time the furthest up-current detection is set as the location of the source. The Post Declaration Maneuvers were designed to execute multiple passes around the declared source location allowing additional sensors to gather data on its location.

For the Reacquire behavior, the AUV conducts a maneuver depicted in Figure 2.22. This "bow tie" maneuver is executed at N points on the detection list. This allows a maximum of N "bow tie" maneuvers to occur and provide ample opportunity to reacquire to plume.

The authors state that the AUV successfully navigated the chemical plume from over 975 m away with an average source detection accuracy of 13 m. One set of experiments yielded 7 successful runs out of 8. This research reveals a handful of potentially effective odor based navigation schemes for various situations an AUV (or UAV) may encounter in the 2 dimensional realm. However, there was a lack of detail in the discussion of the experimental results. This causes some concern to the overall robustness and validity of the navigation algorithms used. The manner in which the material is presented leads the reader to believe the AUV maneuvers were based on

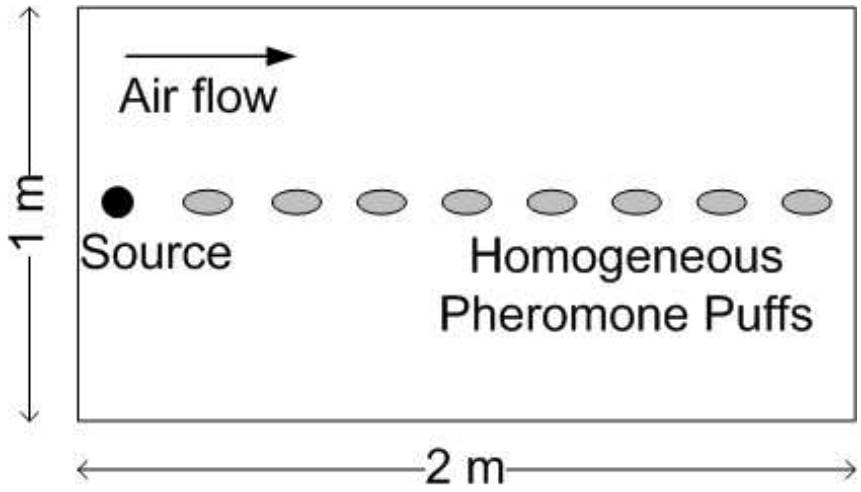
a detection by detection basis. In other words, as soon as the dye was detected the AUV makes a maneuver decision and the following detection would cause another maneuver decision. With a detection rate of 10 Hz, this seems to be too fine of a scale for high dynamic maneuver decision making.

2.3.6 Belanger and Arbas Implementation. Belanger and Arbas [5] developed three, two-dimensional, odor-based navigation algorithms based on various hypothesis of how male moths navigate pheromone plumes. As biologists, they were not interested in maximizing the performance of the algorithms. Instead their goal was to test the existing hypotheses to see how close to a real moth they performed. As such, they were limited in their algorithm development by the moth's physical and cognitive abilities. The truth data used for comparison purposes was taken from high speed camera footage of a birds-eye-view of wind-tunnel experiments observing live moths navigating a true pheromone plume to its source.

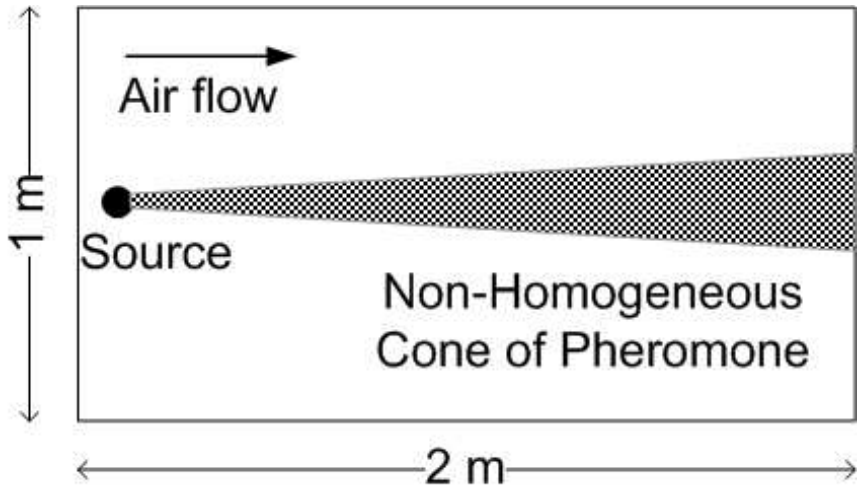
The simulation environment described was designed to duplicate the environment produced in the wind tunnel experiments.

1. 1 m $\times$ 2 m operational area
2. Pheromone source was located 2 m downwind from moth's starting location
3. 100 $\frac{\text{cm}}{\text{s}}$ wind speed with 5% random variation
4. Wind direction was constant

The airflow in the wind tunnel was laminar. This streamlined flow caused minimal variation in the plume's structure, allowing for simplistic plume models to be used in the simulations. The navigation algorithms were tested against three different plume structures: non-homogeneous cone (10 cm wide), homogeneous puffs (0.14 s in duration and emitted at a 5 Hz rate), and a static image of a smoke plume. Figure 2.23 depicts the appearance of the first two plumes while the static smoke plume can be seen in [5].

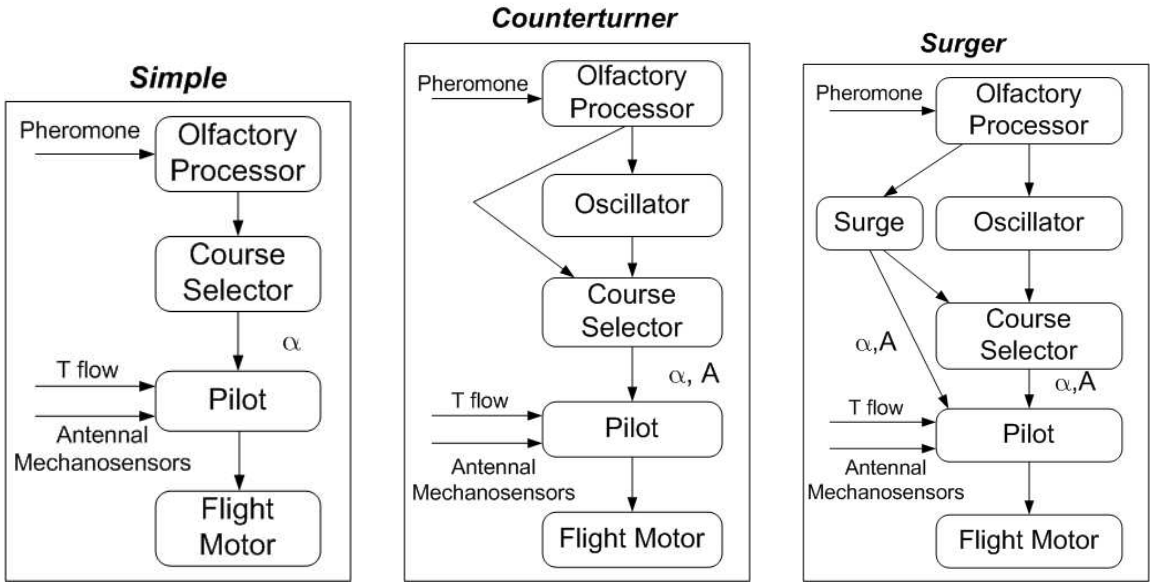


(a) Homogeneous Puff Plume



(b) Non-Homogeneous Cone Plume

Figure 2.23: (a) Homogeneous plume with puffs emitting at a rate of 5 Hz and 0.14 s in duration. (b) A non-homogeneous cone shaped plume (This figure was reproduced from [5]).



(a) Simple Algorithm (b) Counterturner Algorithm (c) Surger Algorithm

Figure 2.24: (a) Simple algorithm. (b) Counterturner algorithm. (c) Surger algorithm (This figure was reproduced from [5]).

The navigation algorithms were the only part of the simulations which differed. However, there were many identical subroutines used in each of the three navigation schemes. Flow charts of the three navigation algorithms are displayed in Figure 2.24, showing the incorporation of the common subroutines. The subroutines used amongst all three simulations are:

1. **Sensor**: a 2 cm circle which acts as the pheromone detector
2. **Course Selector**: produces new heading angles and velocities
3. **Pilot**: given new heading and velocities, compute a change in velocity and change in heading to execute
4. **Flight Motor**: propagate simulation forward each time step

Success of each simulation was based on the simulated moth reaching to within 10 cm of the pheromone source under a specified time limit. The three navigation algorithms simulated were:

1. **Simple**: As discussed in the biology section of this chapter, there is no debate that when moths detect pheromone they modulate their flight upwind in a sinusoidal type pattern. The Simple algorithm uses this behavior as its only means of navigation. When pheromone is detected the simulated moth randomly sets a new course heading within +/- 60° of the wind-line. The maneuver is continued, ignoring pheromone, until the new course is achieved. This algorithm had a success rate of only 10%. However, when it did succeed it did so in the quickest time of any navigation algorithm with an average time of 5.1 +/- 0.2 s.

2. **Counterturner**: This algorithm incorporated the hypothesis that MSexta has an internal timer which causes the temporal regularity observed between counter turns. Oscillator is a subroutine employed to mimic this behavior. If pheromone is detected, a clock is activated to keep track of the time since a detection last occurred. Course Selector is then activated to conduct turns at a regular rate of 450 ms. If the time since detection is within a given threshold, the commanded heading angle range from +/- 50° off the wind-line. If the time since detection exceeds the threshold, the commanded heading is $+/-85°$ to 105° off the wind-line in an attempt to reacquire the plume. Although the flight paths produced by Counterturner exhibited moth like characteristics, it still only achieved a success rate of 20% to 30%, depending on plume structure. However, given a longer simulation time of five minutes, more than 50% of trials succeeded in reaching the source.

3. **Surger**: This model was based on work done by Baker and Vickers [3,5,44]. An additional hypothesis suggested contact with the plume causes a suppression of the internal timing mechanism and motivated a surge upwind. This surge consisted of heading angles approaching 0° toward the wind-line. Latency between the detection of the odor and beginning of the surging behavior was observed in Heliothis Virescens to be 300 ms and a surge duration of 380 ms [5, 44]. Since

no such data was available for MSexta, the authors used this data as a starting point.

Along with implementing the Oscillator subroutine for the temporal timing hypothesis, a surge subroutine was incorporated to mimic the additional surge behavior discussed above. After pheromone had been detected, the simulation conducts its normal counterturning behavior (as in Counterturner). When a given surge latency expires, the simulation alters its commanded course angle to +/- 25° off wind-line for a given length of time, known as the surge duration. If pheromone is detected during the surge, then another surge is executed, otherwise the casting flight associated with the oscillator subroutine (identical to Counterturner) resumes.

The flight profiles produced by Surger did not accurately represent a moth type behavior. The success rate was also poor, with only about 2% reaching the source in the allotted time.

Although the results of the algorithms above were not spectacular, especially given the 70% success rate of MSexta wind tunnel tests [5], useful information was still gained. The Counterturner algorithm provides the most useful conclusions. It seems as though heading angles of around +/- 40° to the wind-line produce the most successful counterturning behavior. Also, the way the authors incorporated the detections causes some concern. After a maneuver is completed the simulation uses the next, single pheromone detection to trigger a new maneuver. The decision to change behavior based on every sample seems as though it would be too noisy of a process. Further discussion on this matter is left for Chapter III.

2.4 Summary

The discussion on the abilities of moths, especially MSexta, to navigate pheromone plumes gives a solid foundation for the development of an algorithm designed to track a chemical plume in the horizontal plane. The simulation and robotic implementation

section gave insight into methods currently used in tracking chemical plumes in the horizontal plane. A summary of these techniques are given in Table 2.2. Some of these methods are used to aid in the development the navigation algorithm discussed in Chapter III. However, the task of tracking a chemical plume in the vertical plane, as needed for 3-D simulations, has been absent from most research in the open literature. Therefore, these techniques had to be engineered in an ad hoc manner and are also discussed, in detail, in Chapter III.

Table 2.2: Summary of existing odor-based navigation algorithms.

Implementation	Moth based	Bacteria based	Gradient search	Fish based
Li	X			
Marques	X	X	X	
Vabø		X	X	X
Ishida	X			
Farrell	X			
Belanger	X			

III. Simulation Development

The biological inspiration for the odor-based navigation algorithms developed are derived, primarily, from the behaviors of male moths tracking female pheromone plumes. The research conducted on MSexta, as discussed in Chapter II, was used extensively [2, 5, 50]. Toward the goals of this thesis, two navigation algorithms were developed. Each algorithm has its own simulation environment: a 2-D "moth-like" simulation and a 3-D UAV simulation. The difference between the simulations, apart from the dimension, is the moth simulation is constrained by the physical and cognitive abilities a moth possesses, while the UAV simulation is limited by the abilities of a small UAV. While the two simulation environments have different capabilities, they share the following components.

1. A time-varying chemical plume the air vehicle is required to track.
2. A sensor model onboard the aircraft used to detect the airborne chemical.
3. The bio-inspired navigation algorithm used to make maneuver decisions based on sensor detections.
4. A dynamics model designed to mimic capabilities of either a moth or UAV.

A flowchart of the simulation is given in Figure 3.1. The major components are discussed in detail in the following sections.

3.1 2-D "Moth-like" Simulation

As previously discussed, this simulation is designed to mimic a male moth's pheromone tracking capability. In doing so, one must conform to the moth's physical and cognitive abilities. Most of the research in the open literature examines only the moth's behavior in the horizontal plane. When the moth makes contact with the plume, it maintains a stable altitude; especially within the confines of the wind tunnel experiments mentioned in [5]. This was the reason for keeping the simulation in 2-D: attempting to mimic the moths behavior as seen in wind tunnel experiments. As such, this simulation was limited to a 1 m $\times$ 2 m (width $\times$ length) area of operation.

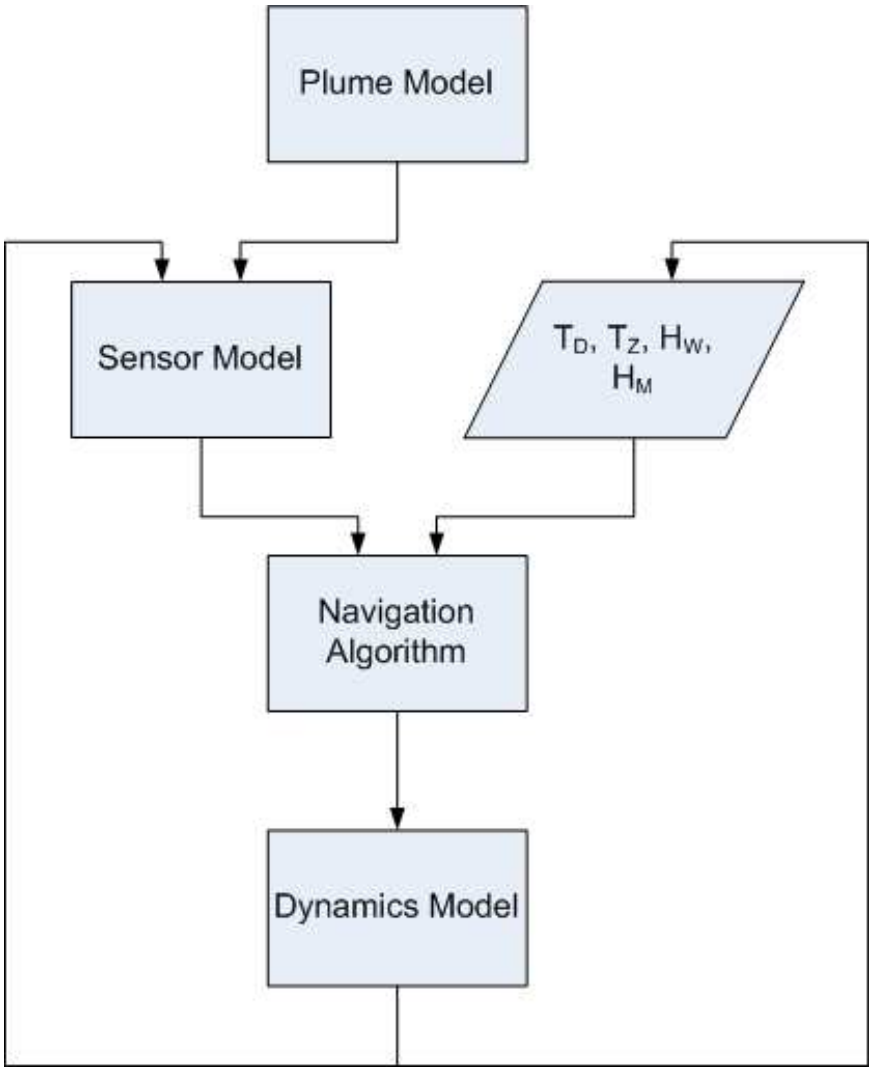


Figure 3.1: Top level flow chart of the simulation. T_D = Time since last plume detection, T_Z = Time since last crossed the wind line, H_W = Wind heading, H_M = Moth heading.

A critical part of any odor-based navigation simulation is the development of a realistic, dynamic chemical plume. The navigation algorithms would be limited in their usefulness to real-world applications without such models. Since this forms the foundation from which the simulation is based, the plume model is discussed first. The moth's dynamics model follows, as does the sensor model, and navigation algorithm.

3.1.1 Plume Model. Methods of modeling the dispersion of chemicals in a wind field often presented in the open literature [8, 15, 28, 31, 34, 37, 39, 42, 46] include the Eulerian, Lagrangian, and Eulerian-Lagrangian techniques. The Eulerian has been found less accurate than the stochastic-based Lagrangian model [8]. However, the Eulerian approach remains more computationally feasible than the Lagrangian [8]. The Eulerian-Lagrangian model uses the strengths from both to create an accurate and computationally-feasible model. This approach uses the Eulerian method to model the flow while the Lagrangian method is used to calculate the coordinates of all the particles dispersed in the flow field [8].

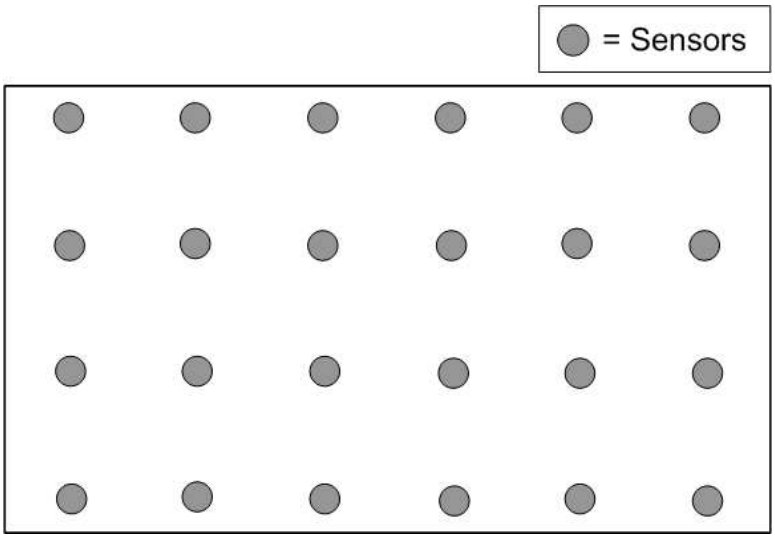
3.1.1.1 High Fidelity Model. The Eulerian-Lagrangian method was used by Dr. Wayne C. Jouse in a chemical dispersion model available for use with this research effort. This model has the ability to generate 2-D as well as 3-D plumes. A basic description of the model is:

1. An area of operation is defined with sensors placed at user-defined intervals within the said space. Figure 3.2 illustrates how the space and sensors appear in both 2-D and 3-D. The sensors are the location where wind field data and concentration data are calculated. Using this method, the values for areas between sensors can be found by interpolation.
2. Given user-defined atmospheric conditions and chemical source conditions, a single filament of chemical is transported through the entire area of operation. Figure 3.3 illustrates this concept in 2-D, showing the turbulent zones and filament path with associated velocity vectors.

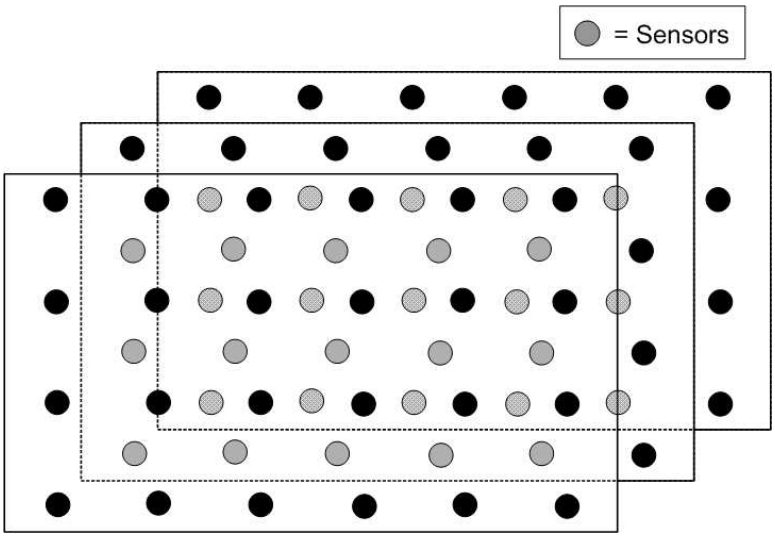
3. The path of this single filament defines the centerline of the plume. Given the atmospheric conditions across the area of operation, the advection-diffusivity of the single filament is calculated at every sensor location. This assigns a concentration level to each sensor which defines the plume structure. Figure 3.4 gives an example of a 2-D and 3-D plume generated by this program. Red represents the highest concentration while blue represents the lowest. Notice the uniform spacing of the sensors.

4. Once the plume has been propagated over the range of the area of operation, as in Figure 3.4, the same procedure must be completed for every time step. For this research a time step of 10 Hz is needed over simulation lengths of up to 7,000 s.

This algorithm develops the entire time-history of a chemical plume and cannot be run in conjunction with the navigation simulation. In other words, it is not computationally feasible to propagate the plume at the same time the simulated moth or UAV is being propagated. An extensive amount of time was needed for this algorithm to generate just a subset of plume data needed for a complete simulation. Approximately 18 hours were required to generate less than 1,000 seconds of data. Given the amount of plume data needed for this research (a minimum of 40 different plume structures over time spans of up to 7,000 s), this algorithm was deemed too complex to be used for running the simulations needed for this thesis. However, the subset of 2-D and 3-D data generated in trial runs of this software were ideal as a basis for which to test the preliminary navigation algorithms, hence the discussion of this program. The two plume structures used for the 2-D and 3-D preliminary navigation simulations are given in Figure 3.5 and do not change with time. In other words, they are static. These plume structures are the same as in Figure 3.4, except with noise added to the position of the sensors. As the sensor is used to represent the location of a chemical particle, and particles would never be uniformly spaced, the addition of noise made the plume structure appear more realistic. Such plume structures also



(a) 2-D Sensor Map



(b) 3-D Sensor Map

Figure 3.2: Depiction of sensor maps used in plume development. (a) 2 dimensional sensor map. (b) 3 dimensional sensor map.

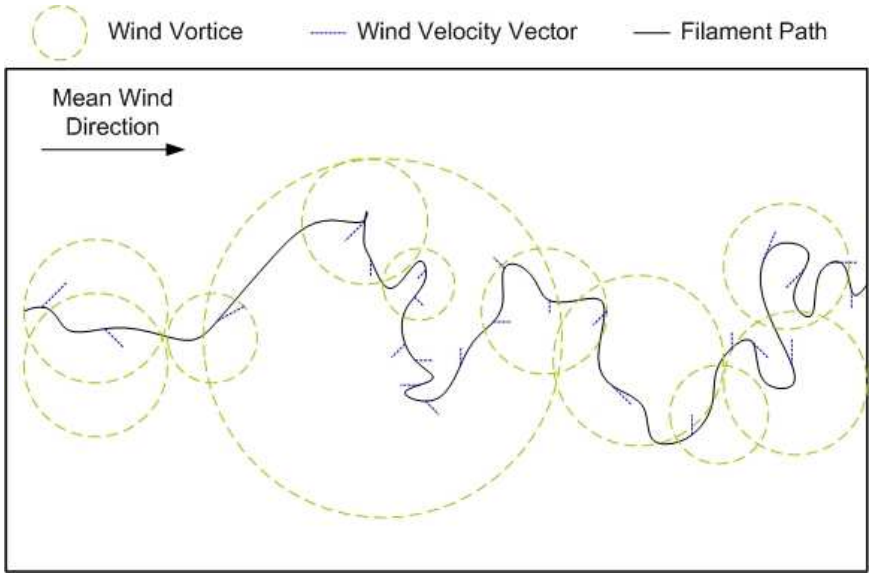
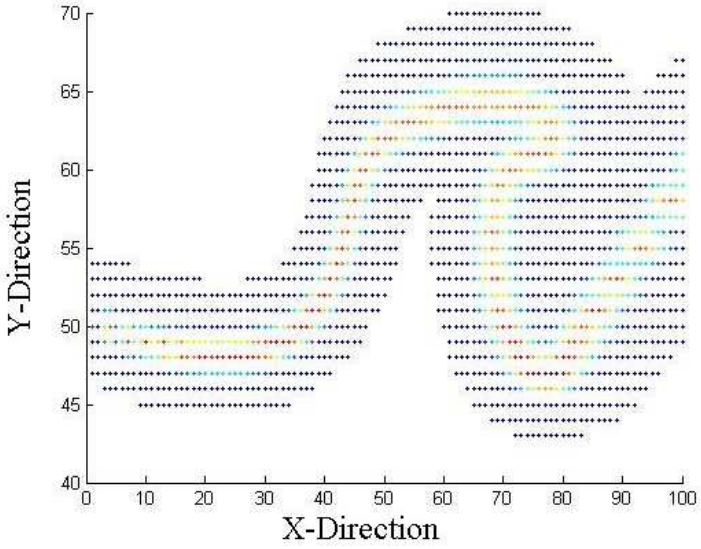


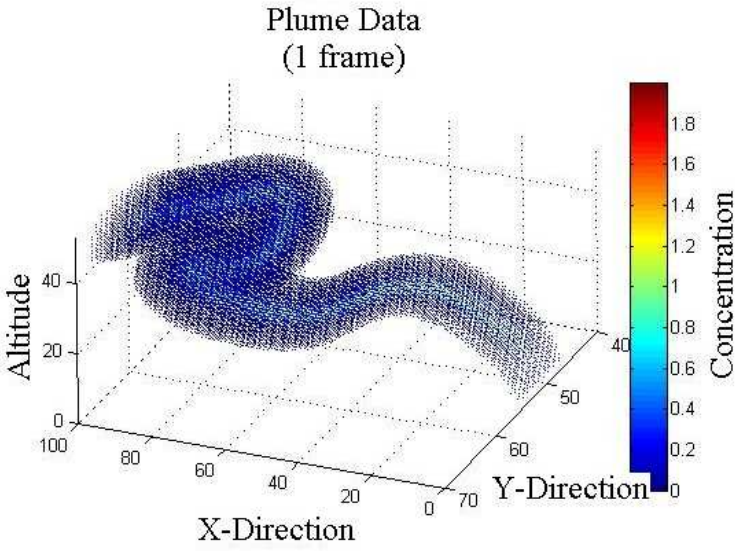
Figure 3.3: The filament is propagated through the area of operation by the mean flow of the wind and the wind vortices of varying sizes and strengths.

serve as the point of reference in comparing the lower fidelity, more computationally efficient plume generation algorithms.

3.1.1.2 Low Fidelity Model. The literature on odor-based navigation algorithms reviewed for this thesis effort consisted of using binary sensors for detecting the airborne chemical [5, 12, 16, 21, 29, 43]. The use of concentration detected by a moth plays a useful role in its navigational abilities. However, if the concentration of particles in a simulated plume can be ignored, the computation time of such a plume would be significantly decreased. These two factors led the development of a lower fidelity simulated plume, containing just particle position with no concentration information. This low fidelity plume model was simplified further by not generating sensors over the entire area of operation and not independently creating wind vortices. A simpler method can be used, other than those given by the Eulerian, Lagrangian, and Eulerian-Lagrangian methods, to provide a simulated plume with sufficient meandering and spatial characteristics representative of the high fidelity plume. This

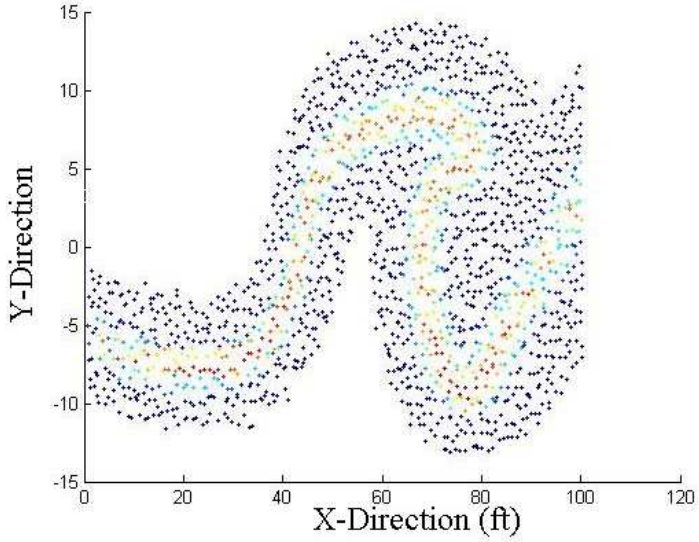


(a) 2-D Plume

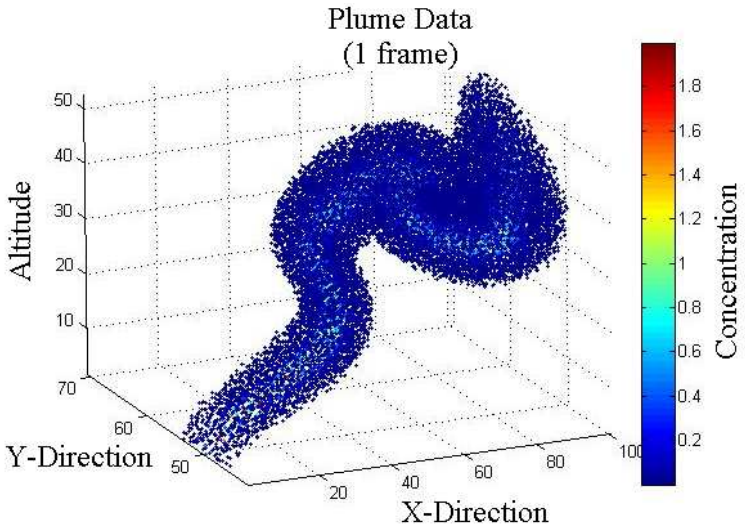


(b) 3-D Plume

Figure 3.4: Static plumes used in early development of 2-D and 3-D navigation algorithms. (a) 2 dimensional plume. (b) 3 dimensional plume.



(a) 2-D Plume



(b) 3-D Plume

Figure 3.5: Static plumes with noise added to sensor positions. (a) 2 dimensional plume. (b) 3 dimensional plume.

assessment of the low fidelity plume is a purely qualitative. The method developed to generate this lower fidelity plum is as follows:

1. The density of the plume does not need to be accurate. The varying sizes of the simulation's sensor are used to mimic high, medium, and low density plumes. This is discussed further in Section 3.1.3.
2. The plume algorithm developed generates a new particle from the source location every 100^{th} of a second. The particle's initial position is randomly chosen within a 1 mm $\times$ 1 mm square. Each particle is monitored independently, until a maximum of 2,000 particles have been generated. The oldest particle is deleted as new ones are generated.
3. Each particle is propagated to a new position every 100^{th} of a second, . This propagation is a stochastic process, depending on the wind's direction and velocity variances. The equations of motion for the particles are given by:

$$H_P = 2\pi\varrho(0,1) \tag{3.1}$$

$$V_P = V_W + 0.5\eta(0,1) \tag{3.2}$$

$$V_{P_x} = V_P\cos(H_P) + D_{P_x} \tag{3.3}$$

$$V_{P_y} = V_P\sin(H_P) + D_{P_y} \tag{3.4}$$

$$X_P(t) = X_P(t-\Delta t) + V_{P_x}\Delta t \tag{3.5}$$

$$Y_P(t) = Y_P(t-\Delta t) + V_{P_y}\Delta t \tag{3.6}$$

Where

(a) $H_P \equiv$ Particle heading (plume particle) (deg)

(b) $V_P \equiv$ Particle velocity ($\frac{\text{m}}{\text{s}}$)

(c) $V_{P_x} \equiv$ X component of particle velocity ($\frac{\text{m}}{\text{s}}$)

(d) $V_{P_y} \equiv$ Y component of particle velocity ($\frac{\text{m}}{\text{s}}$)

(e) $X_P \equiv$ X component of particle position (m)

(f) $Y_P \equiv$ Y component of particle position (m)

(g) $D_{P_x} \equiv$ X component of particle drift ($\frac{\text{m}}{\text{s}}$)

(h) $D_{P_y} \equiv$ Y component of particle drift ($\frac{\text{m}}{\text{s}}$)

(i) $t \equiv$ Time (s)

(j) $\Delta t \equiv$ Sample Time (s)

(k) $\varrho(0,1) \equiv$ Uniform random variable between 0 and 1

(l) $\eta(0,1) \equiv$ Normal random variable with mean of 0 and variance of 1

4. An initial plume is generated from 2,000 particles that can then be propagated over time via the stochastic process mentioned above. One example of an initial plume is illustrated in Figure 3.6 with final plume shown in Figure 3.6. The same set of equations and methodologies are applied to the simulation so that the initial plume can be propagated in real during the simulation. This works well, as a Monte Carlo simulation of 100 2-D runs takes approximately 75 minutes to complete and the initial plume used to start the simulation takes fewer than 5 minutes to generate.

The description provided here is for the 2-D plume model. The 3-D model, which is similar, is described in Section 3.2.1.

3.1.2 Dynamics Model. Before discussion of the navigation algorithm takes place, the dynamics model must be defined. One must know what parameters the navigation algorithm needs to generate, as these drive the dynamics model. Due to the low weight and the incredible maneuverability of a moth, as discussed in Section 2.1 of Chapter II, a point mass, coordinated turn model was chosen. Equations (3.8) through (3.12) are the mathematical representations of the dynamics model [40].

$$V_M = V_{M_x}\cos(H_{M_{rel}}\Delta t) + V_{M_y}\sin(H_{M_{rel}}\Delta t) \tag{3.7}$$

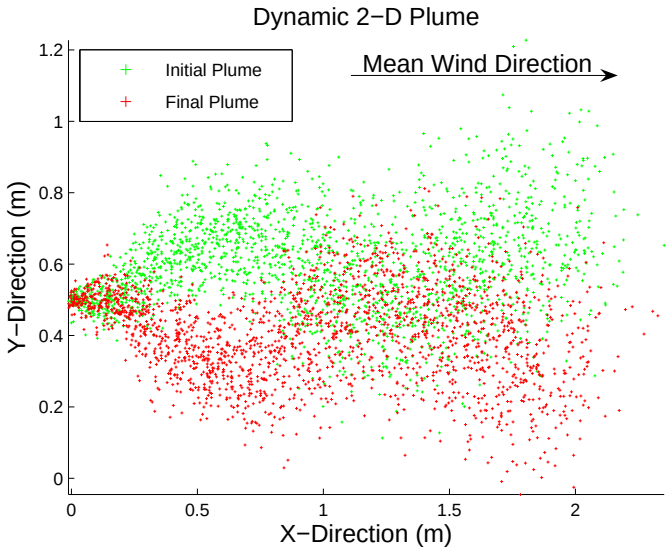


Figure 3.6: Two dimensional chemical plume. Each particle of the chemical plume is propagated through the area of operation by the mean flow of the wind and it's associated variances.

$$sw = \frac{\sin(\omega_M \Delta t)}{\omega_M} \tag{3.8}$$

$$cw = \frac{1 - \cos(\omega_M \Delta t)}{\omega_M} \tag{3.9}$$

$$s = \sin(\omega_M \Delta t) \tag{3.10}$$

$$c = \cos(\omega_M \Delta t) \tag{3.11}$$

$$\begin{bmatrix} \dot{X}_M \\ V_{Mx} \\ Y_M \\ V_{My} \\ \omega \end{bmatrix} = \begin{bmatrix} 1 & sw & 0 & -cw & 0 \\ 0 & c & 0 & -s & 0 \\ 0 & cw & 1 & sw & 0 \\ 0 & s & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_M \\ V_{Mx} \\ Y_M \\ V_{My} \\ \omega_M \end{bmatrix} \tag{3.12}$$

Where

1. $H_{M_{rel}} \equiv$ Moth's heading relative to H_W (deg)
2. $V_M \equiv$ Moth's velocity $(\frac{\text{m}}{\text{s}})$
3. $V_{M_x} \equiv$ X component of moth's velocity $(\frac{\text{m}}{\text{s}})$

4. $V_{M_y} \equiv$ Y component of moth's velocity ($\frac{\text{m}}{\text{s}}$)
5. $X_M \equiv$ X component of moth's position (m)
6. $Y_M \equiv$ Y component of moth's position (m)
7. $\omega_M \equiv$ Moth's turn rate ($\frac{\text{deg}}{\text{s}}$)

The position and velocity variables are a natural part of most dynamics models and Δt always play a role in a discrete time system. The simulations conducted in this research use a $\Delta t = 0.01$ s when dealing with the propagation of the dynamics model. However, the sample time of the sensor is based on the characteristics of MSexta and is discussed in the next section.

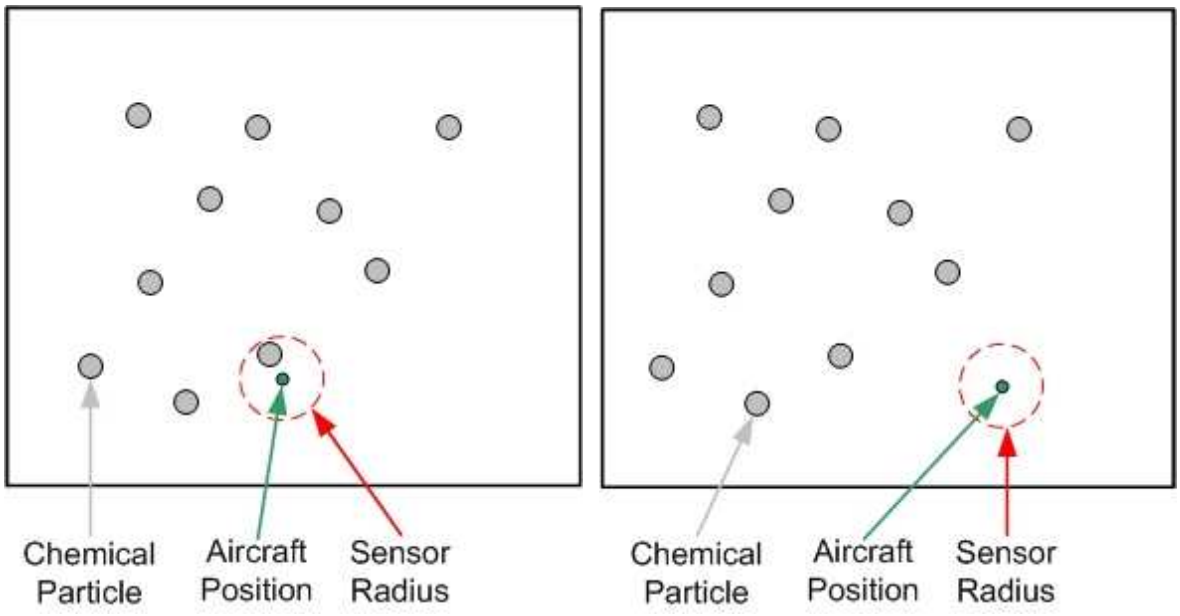
To stay within the flight envelope of a moth, the simulation's velocity, V_M, and ω_M must be bounded to the capabilities of the moth [50].

1. $0 \ \frac{\text{cm}}{\text{s}} \leq V_M \leq 113 \ \frac{\text{cm}}{\text{s}}$
2. $0 \ \frac{\text{deg}}{\text{s}} \leq \omega_M \leq 420 \frac{\text{deg}}{\text{s}}$

The use of ω_M in this model allows for varying the aggressiveness of a turn. A large ω_M results in a sharp turn, similar to those associated with the moth's casting behavior, as depicted in Figure 2.4. A small ω_M allows for a more gradual turn, as with the moth's surging or counterturning behavior, also depicted in Figure 2.4.

One does not typically think of ω_M as an output of a navigation algorithm. Heading and velocity are, on the other hand, reasonable variables to generate from such an algorithm, and heading can be translated into ω_M. This translation is part of the navigation algorithm, to be discussed in Section 3.1.4.

3.1.3 Sensor Model. The sensor model used in this simulation is simple. If a filament falls within a set distance (defined by the user) of the moth, then a detection is made. Since the dynamic plume models used by the simulation do not incorporate concentration, the detection is purely binary. This is a common practice used in most odor-based navigation simulations in the open literature [5, 12, 16, 21, 29, 43]. The



(a) 2-D Sensor: Detection (b) 2-D Sensor: No Detection

Figure 3.7: How the sensor detects an airborne chemical. (a) Sensor detects chemical. (b) Sensor does not detect chemical.

size of the sensor is varied during the sensitivity analysis, discussed in Chapter IV. Due to the inaccuracies of the plume model, changing the size of the sensor is more a reflection of the plume density rather than sensitivity of the sensor. Figure 3.7 illustrates how the sensor declares a detection versus no detection.

The sensor has a sampling rate similar to that of a moth. As discussed in Chapter II, male moths have detection rates up to 33 Hz [4, 7]. However, they can only detect up to 10 $\frac{\text{detections}}{\text{s}}$ [21, 38]. Therefore, the sensor used in this simulation had a sampling rate of 10 Hz. The sensor information is passed into the navigation algorithm, and a new maneuver may be generated.

3.1.4 Navigation Algorithm. As previously discussed, moths do not have the opportunity to learn how to navigate a pheromone plume, it is purely instinctual. Therefore, this navigation algorithm employs reactionary laws, not allowing for any true cognitive abilities. The inputs available for use in this algorithm are the same as those afforded to a real moth, with the exception of wind velocity and pheromone

concentration. As moths operate within a set range of ground velocities, regardless of wind speed, the simulation was simplified by controlling the ground speed directly. This simplification allowed the wind speed and simulation airspeed to be ignored, assuming the wind speeds are kept at reasonable values. The concentration levels of the detected pheromone is an important part of the moth's navigational ability. However, due to the simplified design of the pheromone plume in this research, the use of concentration as an input for the navigation algorithm was not feasible. Hence, this one of the limitations of this algorithm, and subsequently the simulation as a whole.

The following inputs are used by the navigation algorithm to make "decisions" that drive the dynamics model:

1. T_D: The time since the moth last detected the pheromone. This is critical in determining wether the moth should begin to cast in order to relocate the pheromone plume.
2. T_Z: The time since the moth last crossed the wind line. This is helpful in controlling the time allowed between counterturns.
3. d: The binary output of the sensor, 0 = no detection and 1 = detection
4. H_W: The heading of the wind is the foundation to the moth's navigational capabilities. All of its maneuvers are based on the direction of the wind.
5. $H_{M_{rel}}$: Moth's heading relative to H_W.

The outputs of the main portion of the algorithm are:

1. $H_{M_{new_{rel}}} \equiv$ Moth's new heading relative to H_W.
2. $V_{M_{new}} \equiv$ Moth's new velocity.

Using $H_{M_{new_{rel}}}$ and $V_{M_{new}}$ as outputs is a natural way to attack the problem of navigating a plume. However, the dynamics model for this simulation does not use heading as an input, it uses ω_M. This required a transformation, converting $H_{M_{new_{rel}}}$ to ω_M, that is executed in the second part of the navigation algorithm, NAV2.

3.1.4.1 NAV1. This part of the navigation algorithm is the key of the entire algorithm as it dictates $H_{M_{new_{rel}}}$ and $V_{M_{new}}$. A flow chart of NAV1 is given in Figure 3.8. This methodology in many aspects is similar to those developed previously in the open literature [5, 12, 16, 21, 29, 43]. The commonalities are:

1. The moth's ground velocity is much greater during casting then when counterturning or surging.
2. T_D is used to change from a counterturing behavior to a casting behavior, and vice-versa.
3. T_Z is used to maintain the time between turns.

However, the portion of the algorithm for this research that deals with detecting the pheromone is different than any presented in the open literature. Most simulations base a change in behavior on a single detection of pheromone. These simulations typically ignore the output of the sensor until the maneuver is completed. The first detection upon the completion of the maneuver is used to decide a new maneuver. This method of conducting maneuvers based on instantaneous pheromone detections does not provide logical decision-making process. A simple example of why this is not favorable is illustrated in Figure 3.9. If a moth is casting, trying to find the plume, and upon first contact with the plume changes its behavior to counterturn upwind, the moth will likely place itself on the edge of the plume. Such positioning could leave the moth in a precarious situation, as it is guaranteed to cast again due to half of its flight profile being out of the plume. A more logical decision making process is one which incorporates short-term memory in order to make a maneuver decision. This method takes into account one major hypothesis: moths use the information sensed over a finite period of time to help choose a maneuver. The first implementation of this idea incorporated a short-term memory, 1 second in length, made up of 10 samples from the 10 Hz binary sensor. This short-term memory bank is illustrated in Figure 3.10. The mean of the cell locations in which a detection is made provides a

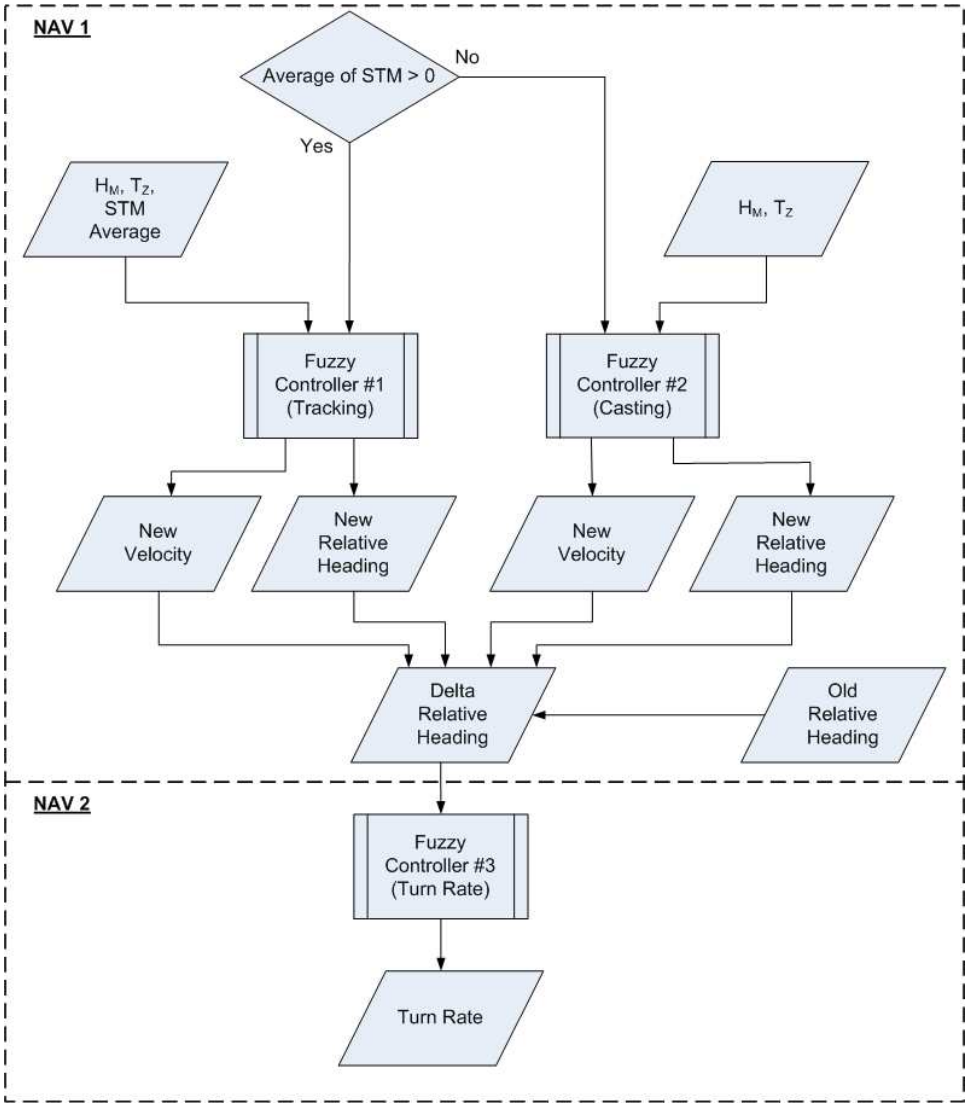


Figure 3.8: Flowchart of 2-D navigation algorithm. The navigation algorithm consists of 2 main sections NAV1 and NAV2 of which the critical components are the three 3 fuzzy controllers.

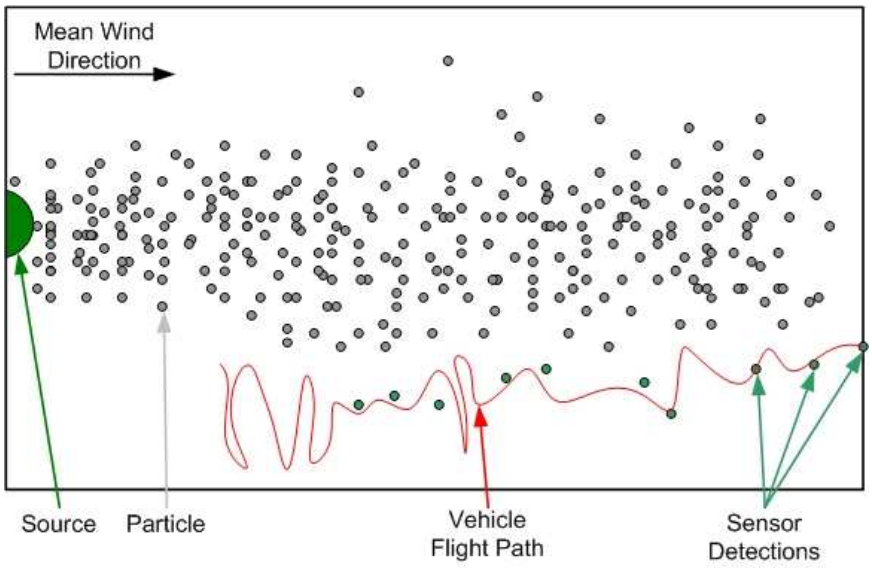


Figure 3.9: Maneuver decision based on single pheromone detection. Deciding to maneuver an aircraft based on a single detection is not the best odor-based navigation method, as it can easily keep the vehicle from entering the plume.

measure of where the moth is located, with regard to the plume:

$$STM_{ave} = \frac{\sum_i Cell_i}{\#\text{ detections}} \tag{3.13}$$

Where

1. $Cell_i$ = Cell location in which a detection is made

The range of STM_{ave} is:

$$STM_{ave} = \begin{cases} 0 & , \quad \text{Out of plume} \\ 1-4 & , \quad \text{Entering the plume} \\ 3-8 & , \quad \text{In the plume} \\ 6-10 & , \quad \text{Leaving the plume} \end{cases}$$

Figure 3.11 gives examples of each of the 4 possible outcomes mentioned above. This decision of the moth's location within the plume is fuzzy. For example, the average value of 3.5 corresponds to both "Entering the plume" and "In the plume." This

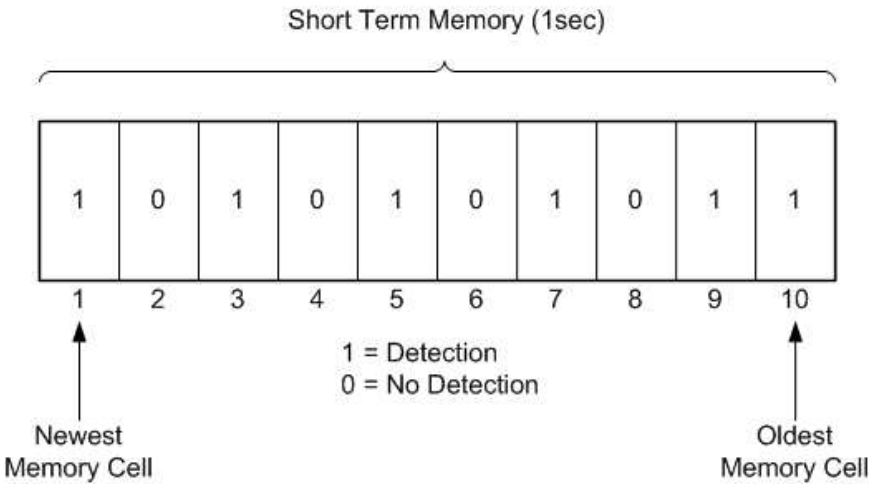
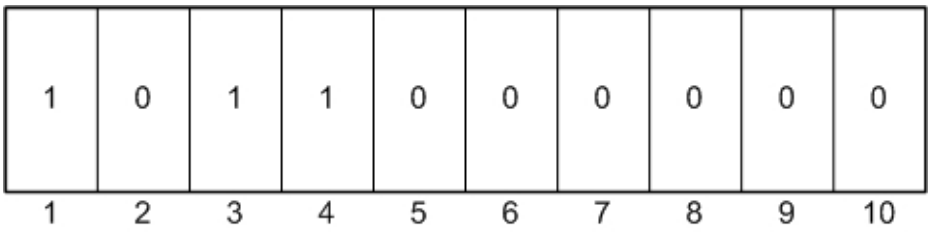


Figure 3.10: Short term Memory. The location of detections in the short term memory dictates where the vehicle thinks it is with respect to the plume.

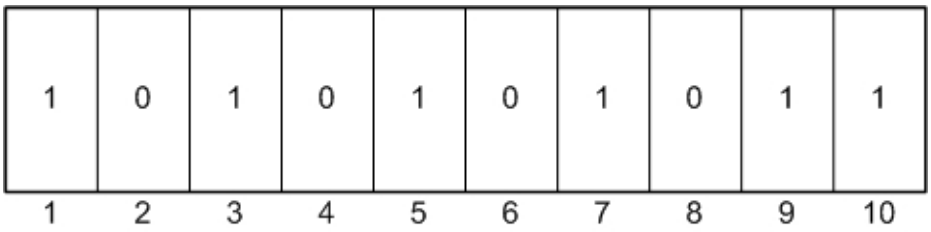
fuzziness of the input memory data was one of the reasons for using a fuzzy controller as the basis for developing the navigation algorithm. The uncertainty, or range, in acceptable values for T_D and T_Z, declared by biological experts in the field also led to the use of fuzzy controllers over more classical control methods.

This algorithm is based on two modes of flight: tracking and searching. The moth's operating mode depends on a detection occurring in the previous second. Specifically, if there are no detections in STM then the "moth" enters search mode, otherwise it enters tracking mode. This is indicative of incorporating a 1 s T_D to switch between modes of flight. The tracking mode elicits maneuvers similar to the counterturning and surging behaviors of moths. While the search mode maneuvers replicate the casting behaviors of moths.

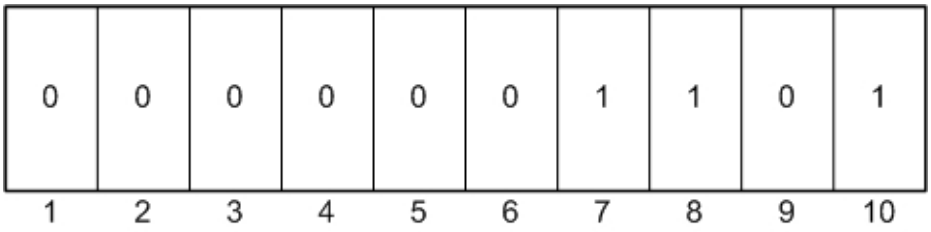
As can be seen from Figure 3.8, the inputs into the tracking fuzzy controller are T_Z, $H_{M_{rel}}$, and STM_{ave}. The associated fuzzy sets are illustrated in Figure 3.12. The membership functions of T_Z were designed with regards to the empirical data discussed in Chapter II. In other words, M.Sexta's average time between turns of 466 to 833 ms was used as the foundation for the T_Z fuzzy set. The membership functions of $H_{M_{rel}}$ reflect the simulation's design of maintaining a +/- 180° relative



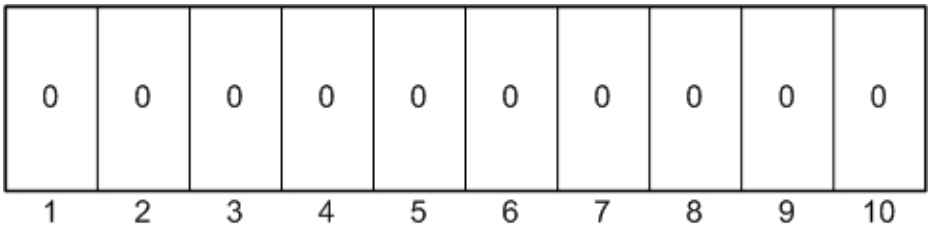
(a) Entering the plume



(b) In the plume



(c) Leaving the plume



(d) Lost the plume

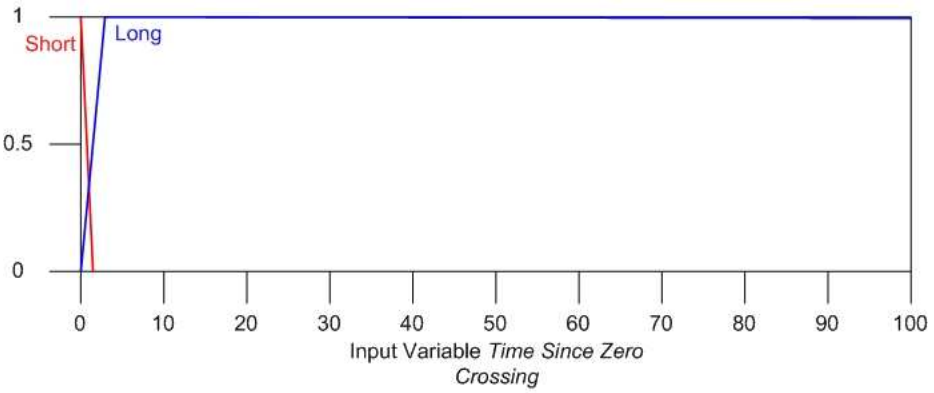
Figure 3.11: Possible short term memory scenarios. (a) Entering the plume. (b) In the plume. (c) Leaving the plume. (d) Lost the plume.

heading angle. These membership functions were roughly centered on the angles which real moths navigate with respect to H_W. The STM_{ave} membership functions were developed from the concepts depicted in Figure 3.11. The output fuzzy sets are shown in Figure 3.13. $H_{M_{new_{rel}}}$ has the same basic meaning as $H_{M_{rel}}$, the heading of the "moth" relative to H_W. Therefore, their fuzzy sets are identical. The membership functions associated with $V_{M_{new}}$ are bounded by the velocity values found in experimental testing as discussed in Chapter II.

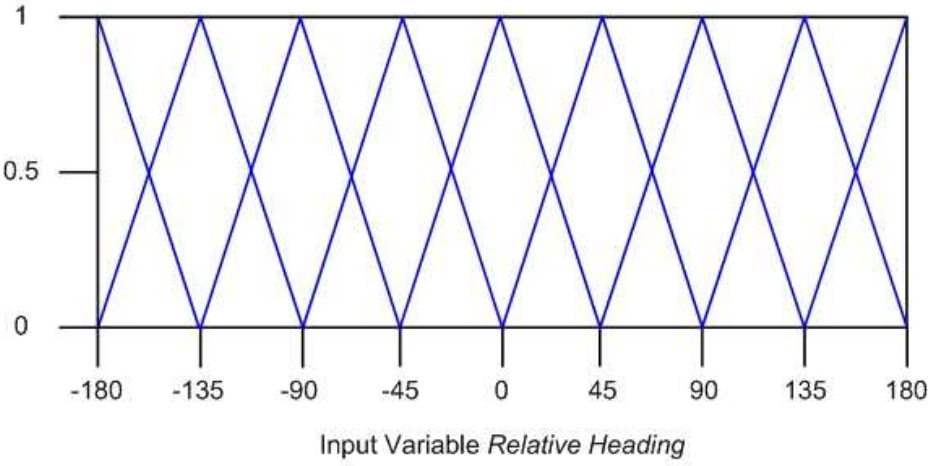
The fuzzy rules, mapping inputs to outputs, were generated from the behaviors mentioned in Chapter II. The rules governing this fuzzy controller are given in Section A.1.1 of Appendix A.

When no detections are in the STM the "moth" switches to search mode. As seen in Figure 3.8, this controller has the same outputs as the tracking controller but has only two inputs, T_Z and $H_{M_{rel}}$. The input fuzzy sets are identical to their corresponding fuzzy sets in the tracking controller. The rule base for this controller sets it apart from the tracking controller. If the "moth" is in search mode, it is searching for the plume; so using the behavior indicative of real moths, the simulated moth travels perpendicular to the wind line. The "moth" makes sharp turns and travels at a high ground speed, covering a larger crosswind distance than when in tracking mode. The rules used in order to replicate this behavior are given in Section A.1.2 of Appendix A.

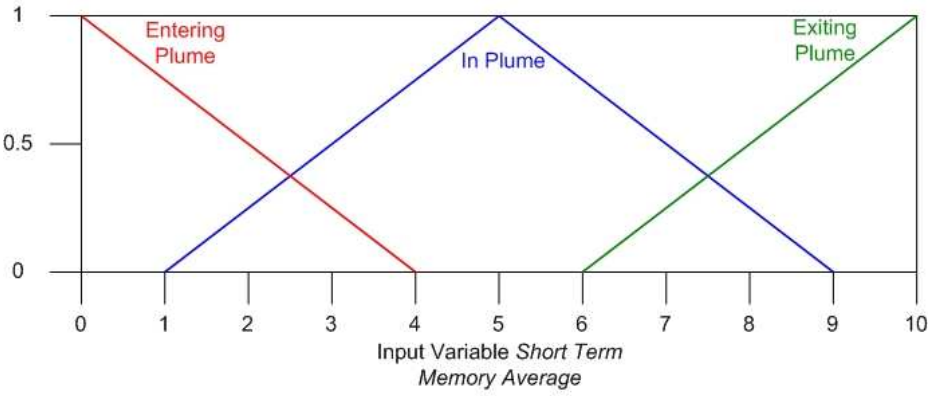
Since both track and search controllers output $H_{M_{new_{rel}}}$, which cannot be used by the dynamics model, NAV2 had to be designed for the purpose of converting $H_{M_{new_{rel}}}$ into ω_M. In other words, the NAV2 algorithm converts the heuristic rules and fuzzy logic commensurate with a moth to the command set needed to control the physical model.



(a) T_Z

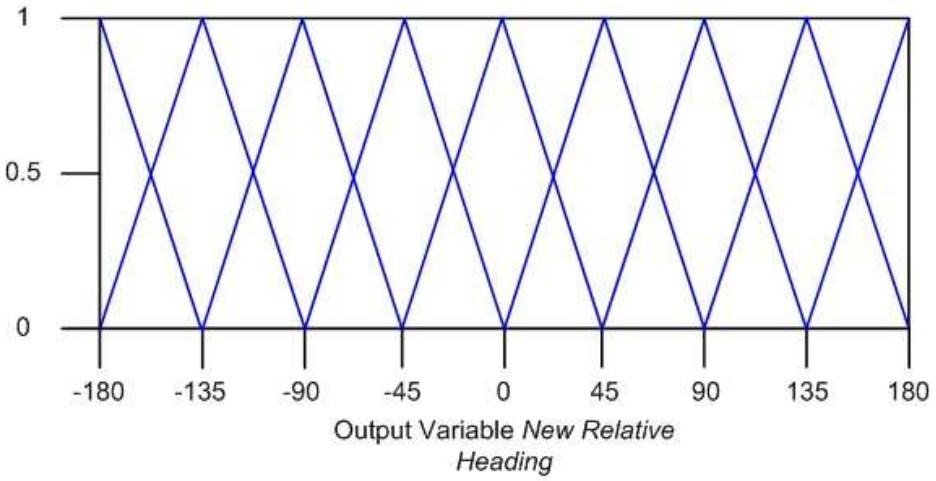


(b) $H_{M_{rel}}$

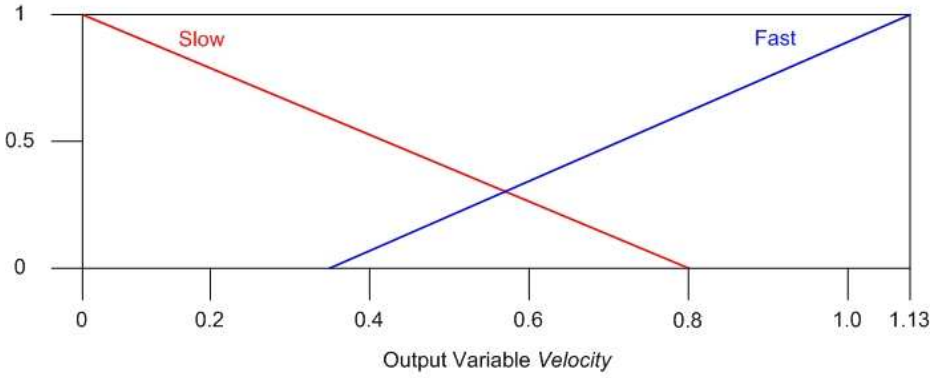


(c) STM_{ave}

Figure 3.12: Input fuzzy sets for 2-D navigation algorithm, NAV1. (a) T_Z. (b) $H_{M_{rel}}$. (c) STM_{ave}.



(a) $H_{M_{new_{rel}}}$



(b) $V_{M_{new}}$

Figure 3.13: Output fuzzy sets for 2-D navigation algorithm, NAV1. (a) $H_{M_{new_{rel}}}$. (b) $V_{M_{new}}$.

3.1.4.2 NAV2. This is a simple fuzzy controller consisting of one input, $H_{M_{delt}}$ given by Eq(3.14), and one output, ω_M.

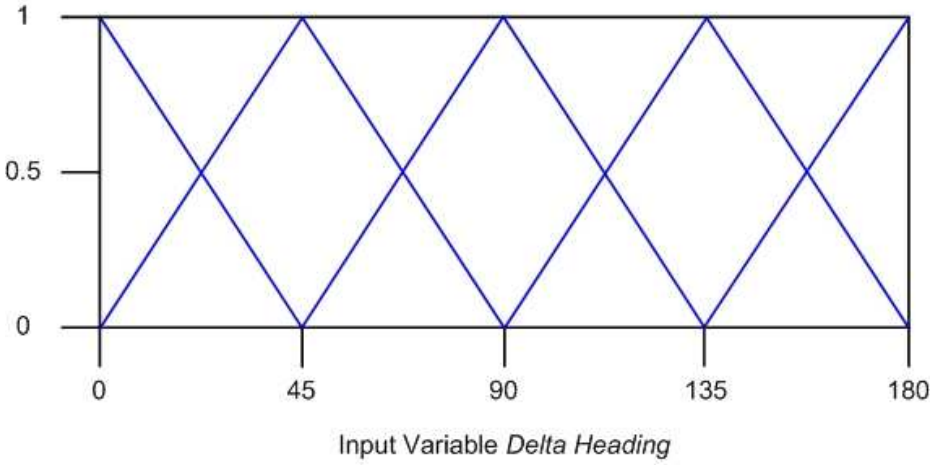
$$H_{M_{delt}} = H_{M_{new_{rel}}} - H_{M_{rel}} \tag{3.14}$$

Fuzzy input $H_{M_{delt}}$ was designed in the same manner as the other heading related fuzzy sets of NAV1 except the absolute value of the delta heading is used. This led to values of the fuzzy set ranging from 0-180°. This range of values is necessary because the simulation is based on a $-180°$ to $180°$ coordinate system. Only the positive values were needed since the magnitude and not the direction of the delta heading is used to determine ω_M. The ω_M fuzzy set was based on the maximum and minimum turn rates of MSexta, as given in the Section 3.1.2. Figure 3.14 illustrates these output fuzzy sets. The rules associated with this controller were derived from the flight profiles of MSexta as discussed in Chapter II. The larger $H_{M_{delt}}$ the faster the turn rate, resulting in a sharper turn. The entire rule set is given in Section A.1.3 of Appendix A.

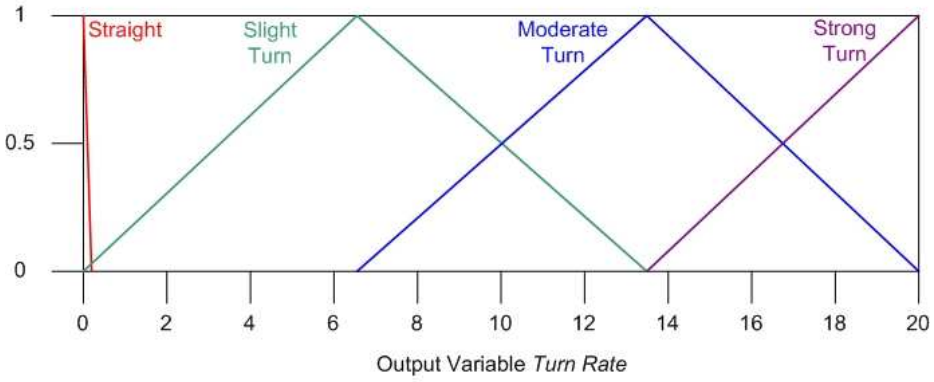
3.2 3-D UAV Simulation

Removing the constraints applied to the 2-D "moth-like" simulation, one is left with a basic scheme on which to base a more complex 3-D navigation algorithm. An inspiration for searching the third dimension, vertical plane, is mostly absent in the open literature. However, one paper, [45], suggests that as the casting flight of a moth expands horizontally, if you look in 3D, they expand vertically as well. Using the knowledge of how plumes are formed in order to generate ad hoc methods for searching within the vertical plane are also used to further address this vertical search problem.

This simulation is of a larger scale than the previous "moth-like" simulation. The area of operation is roughly 10,000 ft × 10,000 ft × 3,000 ft (length × width × height). The term "roughly" is used because the plume will be constrained to this



(a) $H_{M_{delt}}$



(b) ω_M

Figure 3.14: Input/Output fuzzy sets for turn rate fuzzy controller, NAV1. (a) Input fuzzy set, $H_{M_{delt}}$. (b) Output fuzzy set, ω_M.

volume of space. The UAV, however, is only confined to stay above an altitude of 0 ft. This area of operation is free from obstacles and the terrain is flat, therefore, the simulation has 0 ft of elevation.

The area of operation has an associated wind field which can be controlled. The mean direction of the wind can be changed along with its turbulence. This is similar to that of the "moth-like" simulation, except the scale is larger and movement in the vertical plane had to be generated for a 3-D wind field. This is discussed further in Section 3.2.1.

There are four exit criteria for the UAV simulation. The first two are similar to those of the "moth-like" simulation, while the others are unique to the UAV simulation. All are listed below:

1. Time: If the simulation takes longer than 7,000 s, the simulation will end. This time limit was set determined from preliminary testing, providing enough time for the UAV to successfully complete it's mission.
2. Source Located: If the simulation travels within a user-defined distance from the source, the simulation is stopped.
3. Search Pattern Complete: If the simulation completes the final search pattern in an attempt to relocate the plume, without a detection, the simulation will end.
4. Hits the Ground: If at any point in the simulation the UAV has an altitude $\leq$ 0 ft, the simulation will end.

As with the 2-D "moth-like" simulation, and because it lays the foundation from which the simulation is based, the plume model is discussed first. The dynamics model is then discussed, followed by the sensor model and then the navigation algorithm.

3.2.1 Plume Model. As discussed in Section 3.1.1.1, the high fidelity plume model was not a computationally feasible choice for developing a dynamic plume in either 2-D or 3-D. The same methodology used to develop the low fidelity 2-D plume,

of Section 3.1.1.2, was used to generate the 3-D low fidelity plume. This methodology is as follows:

1. As with the 2-D plume, the density of the plume need not be accurate. Variations of the sensor will be used to mimic high, medium and low density plumes. This will be discussed in the Section 3.2.3.
2. The plume algorithm generates one new particle from the source location every second. Each particle's initial position is randomly chosen within a 1 ft $\times$ 1 ft $\times$ 1 ft cube. Each particle is monitored independently, until a maximum of 10,000 particles have been generated. The oldest particle is deleted as new ones are generated.
3. Every 10^{th} of a second each particle is propagated to a new position. This propagation is a stochastic process, depending on the wind's direction and velocity variances. The equations of motion for the particles are based on Newtonian physics. By simply adding a bias to the basic equations of motion, the individual particles can be forced in any direction. In addition, adding noise to each particle causes them to move in a more erratic fashion. This gives the effect of having wind vortices present, varying the density of the plume over it's entire trajectory. The equations of motion for the particles are:

$$H_P = 2\pi\varrho(0,1) \tag{3.15}$$

$$Pch_P = \frac{\pi}{4}\varrho(0,1)\varrho(-1,1) \tag{3.16}$$

$$V_P = V_W + 0.5\eta(0,1) \tag{3.17}$$

$$V_{P_x} = V_P \cos(Pch_P)\cos(H_P) + D_{P_x} \tag{3.18}$$

$$V_{P_y} = V_P \cos(Pch_P)\sin(H_P) + D_{P_y} \tag{3.19}$$

$$V_{P_z} = V_P \sin(Pch_P) + D_{P_z} \tag{3.20}$$

$$X_P(t) = X_P(t - \Delta t) + V_{P_x}\Delta t \tag{3.21}$$

$$Y_P(t) = Y_P(t - \Delta t) + V_{P_y}\Delta t \tag{3.22}$$

$$Z_P(t) = Y_P(t - \Delta t) + V_{P_z}\Delta t \tag{3.23}$$

Where

(a) $Pch_P \equiv$ Particle's pitch angle (deg)

(b) $V_{P_z} \equiv$ Z component of particle's velocity ($\frac{\text{m}}{\text{s}}$)

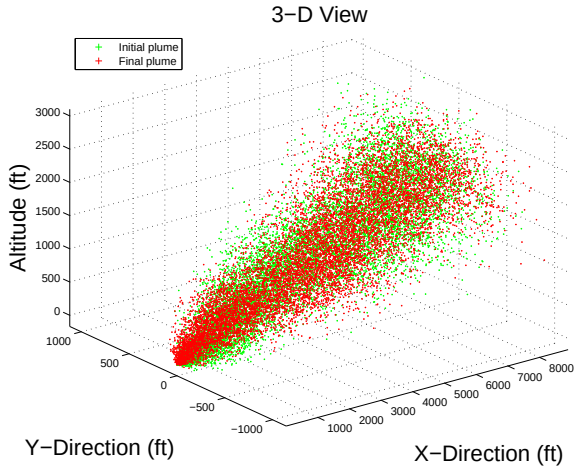
(c) $Z_P \equiv$ Z component of particle's position (m)

(d) $D_{P_z} \equiv$ Z component of particle's drift ($\frac{\text{m}}{\text{s}}$)

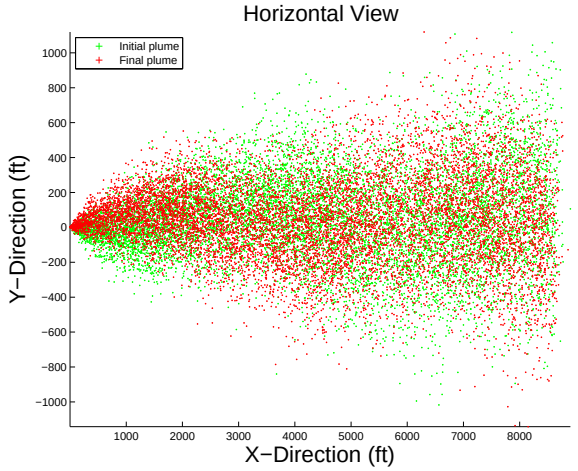
4. An initial plume is generated from 10,000 particles that can then be propagated over time via the stochastic process mentioned above. One example of an initial and final plume is illustrated in Figure 3.6. The same set of equations and methodologies are applied to the simulation so the initial plume can be propagated in real during the simulation. This works well, as a Monte Carlo simulation of 10 3-D runs takes approximately 5 hours to complete and the initial plume used to start the simulation takes fewer than 40 minutes to generate.

3.2.2 Dynamics Model. The goal of this simulation is to aid in the design of an odor-based navigation algorithm for use on a small UAV. This dynamics model should represent a UAV with capabilities and characteristics falling in the following ranges:

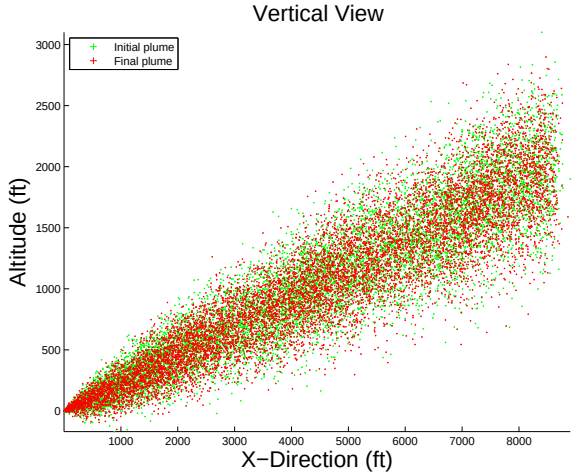
1. Weight: 20 lbs $\rightarrow$ 40 lbs
2. Wing Span: 6 ft $\rightarrow$ 15 ft
3. Speed: 5 $\frac{\text{ft}}{\text{s}}$ $\rightarrow$ 40 $\frac{\text{ft}}{\text{s}}$
4. Altitude: 0 ft $\rightarrow$ 3,000 ft



(a) 3-D Plume, Vertical Plane



(b) 3-D Plume, Vertical Plane



(c) 3-D Plume, Vertical Plane

Figure 3.15: 3-D Low Fidelity Plume. (a) 3-D view. (b) Horizontal view. (c) Vertical view.

If one does not have an accurate dynamics model, the navigation algorithm designed will have less real world applicability. As this thesis is the cornerstone for future hardware implementation research, having an accurate model cannot be overstated.

To meet this need for an accurate dynamics model, a proven 6 degree of freedom, nonlinear aircraft model was used as the starting point for designing the UAV dynamics model. This model is developed in detail in [40], and is based on the dynamics of an F-16. The chosen dynamics model consists of 16 states, the following 12 states and 4 control input states:

1. $V_T \equiv$ Airspeed
2. $\beta \equiv$ Sideslip angle
3. $\alpha \equiv$ Angle of attack
4. $\phi \equiv$ Roll angle
5. $\theta \equiv$ Pitch angle
6. $\psi \equiv$ Yaw angle
7. $P \equiv$ Roll rate
8. $Q \equiv$ Pitch rate
9. $R \equiv$ Yaw rate
10. $p_N \equiv$ Northern position
11. $p_E \equiv$ Eastern position
12. $h \equiv$ Altitude

The control inputs of this model are:

1. $thl \equiv$ Throttle setting
2. $el \equiv$ Elevator deflection
3. $ail \equiv$ Aileron deflection

4. $rdr \equiv$ Rudder deflection

This F-16 model has been proven accurate through unpublished results by Dr. Brad S. Liebst [22]. Dr. Liebst provided the MATLAB® and SIMULINK® programs designed around this model [40]. These programs served as the foundation in developing the 6 degree of freedom UAV model needed for this research.

The UAV model needed has reduced capabilities compared to the F-16 model, and therefore had to be scaled down. The method chosen to scale the F-16 model is based on scaling the mass of the aircraft, m, by the scale factor S. Heretofore, a variable with a hat, " ˆ " , dictates a UAV parameter while no hat dictates an F-16 parameter. To start the scaling process, the equations of mass are used:

$$m = \rho \cdot Vol \tag{3.24}$$

$$\hat{m} = \hat{\rho} \cdot \hat{Vol} \tag{3.25}$$

Where Vol is the volume of the aircraft and ρ is its density. Assuming that scaling has no effect on density

$$\rho = \hat{\rho} \tag{3.26}$$

This allows m to be scaled to $\hat{m}$:

$$\hat{m} = \frac{m}{S} \tag{3.27}$$

Therefore,

$$\hat{m} = \frac{\rho \cdot Vol}{S} \tag{3.28}$$

Since volume is a function of length

$$Vol = \ell^3 \tag{3.29}$$

Equations(3.24) through (3.29) are used to find how the scaling of m by ς effects the scaling of ℓ.

$$\frac{\rho \cdot Vol}{S} = \hat{\rho} \cdot \hat{Vol} \tag{3.30}$$

$$\frac{Vol}{S} = \hat{Vol} \tag{3.31}$$

$$\frac{\ell^3}{S} = \hat{\ell}^3 \tag{3.32}$$

Thus

$$\hat{\ell} = \frac{\ell}{\sqrt[3]{S}} \tag{3.33}$$

Where $\xi = \sqrt[3]{\varsigma}$ denotes the scaling factor of ℓ. Because moments of inertia, $M = \frac{1}{m \cdot \ell^2}$, are functions of both m and ℓ, they will be scaled accordingly:

$$\hat{M} = S \cdot \xi^2 \cdot M \tag{3.34}$$

The last variable to be scaled in the dynamics model is the thrust, T. Since T is a function of mass, it is scaled in the same manner:

$$\hat{T} = \frac{T}{S} \tag{3.35}$$

Given that an F-16 weighs approximately 24,000 lbs and 24 lbs is within the range of acceptable values for the mass of a UAV, a scale factor of $S = 1000$ was conveniently chosen. However, when this dynamics model was tested, the operating envelope was too small to be used in this simulation. In order to increase the operating envelope the scale factor ξ was empirically changed to:

$$\xi = 7\sqrt[3]{S} \tag{3.36}$$

This increased the wingspan of the 24 lbs UAV to 26 ft, exceeding the size of most UAV's of comparable weight. However, this gave flight characteristics which were

within the acceptable ranges previously mentioned. This UAV dynamics model met the needs of this simulation by maintaining stable flight for:

1. Velocities: 10 $\frac{\text{ft}}{\text{s}} \rightarrow 35 \frac{\text{ft}}{\text{s}}$
2. Altitudes: 0 ft $\rightarrow$ above 3,000 ft

Given the inputs available to the dynamics model, three autopilot control loops were used to pilot the UAV: altitude hold, velocity hold, and heading hold. Hence, the navigation algorithm was designed to output a desired altitude, velocity (V_{uav}) and heading (ψ_{uav}). The control loops were designed in SIMULINK® in order to easily interface with the existing software provided by Dr. Liebst. A diagram depicting how the SIMULINK® program functions is given in Figure 3.16. In the discussion of the autopilot commands to follow, the use of a generic "gain" term is used liberally. These gains are typical feedback control gains, adjusted to maintain stability of the control system. In addition, they can be set to give varying levels of feedback performance. In other words, the response time of the auto pilot controls can be increased or decreased depending on the gain setting. Ad hoc tuning of these gains lead to satisfactory response times for the autopilot control loops. The methodologies behind how the autopilot maintains a commanded h, V_{uav}, and ψ_{uav} are:

1. Altitude Hold: Given a commanded altitude, the autopilot measures the difference between the commanded altitude and the current altitude. The feedback loop uses this information in conjunction with the aircraft's current pitch rate, pitch angle, angle of attack, and trim conditions. Only one set of trim conditions were used during the simulations, and were generated for the aircraft traveling at 25 $\frac{\text{ft}}{\text{s}}$ and at an altitude of 1000 ft. These trim conditions, along with appropriate gains, gave the UAV satisfactory performance over the necessary flight envelope. The value generated by the above variables controls the degree to which the elevators are deflected. Figure 3.17 illustrates the flow of this process. The altitude hold capabilities of the UAV can be seen in Figure 3.18. The aircraft stays within a couple feet of the desired altitude during straight and

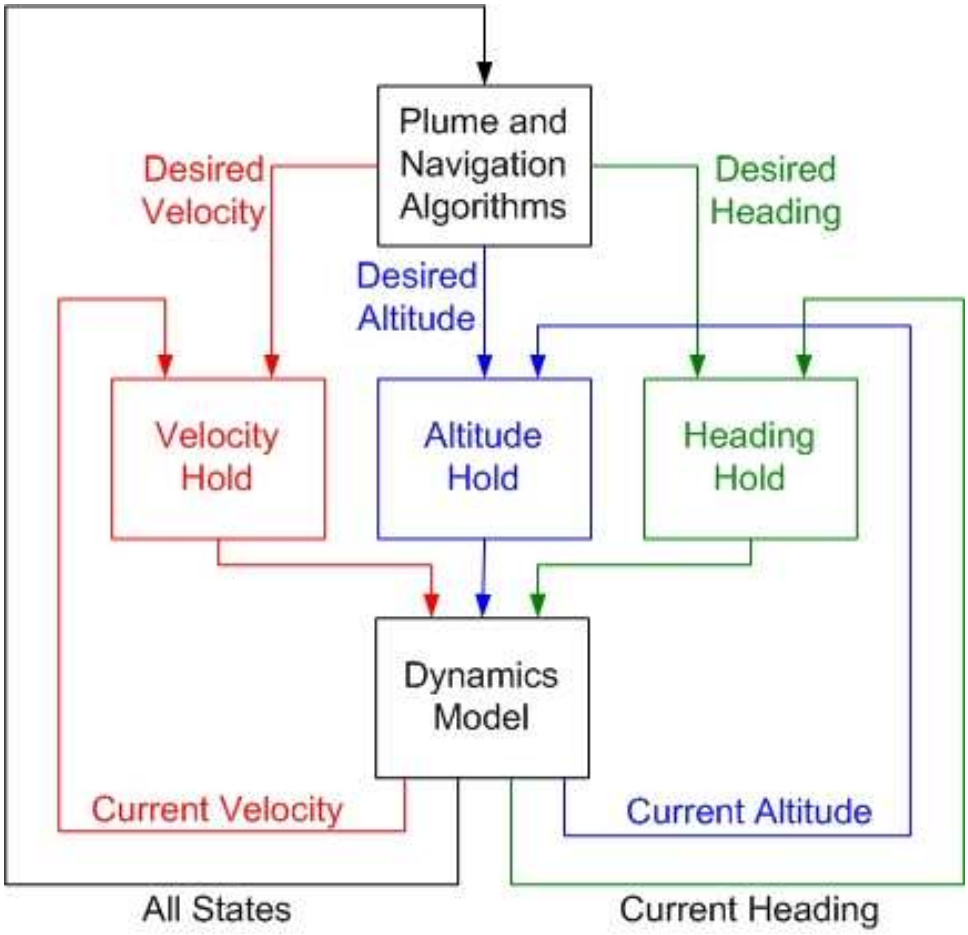


Figure 3.16: Structure of Autopilot SIMULINK® program. Interaction between the different components of the simulation.

level flight. When maneuvering up to 180°, the UAV only drops a maximum of 25 ft and recovers quickly, meeting the requirements of the simulation.

2. Velocity Hold: The autopilot tries to maintain any given commanded velocity, similar to the altitude hold logic. The difference is taken between the commanded and current velocity. The size of this difference dictates how to modify the throttle; increasing or decreasing the speed of the aircraft in an attempt to minimize the difference between the commanded and actual velocity. Figure 3.19 illustrates the details of how this function of the autopilot works. The velocity hold capabilities of the UAV are shown in Figure 3.20. The aircraft is able to hold the velocity within a reasonable tolerance in straight and level flight. The aircraft is also able to recover its desired velocity after making an ascent or descent.

3. Heading Hold: The heading hold function of the autopilot has a similar structure as the velocity and altitude hold functions. In this feedback routine, the heading of the aircraft is assumed to be the yaw angle. An attempt to maintain zero sideslip makes this correlation possible. This function begins by taking the difference between the commanded heading angle and the current yaw angle. Also, roll rate and roll angle are used to aid in the calculation of the aileron control variable. This control variable drives the ailerons in order to correct the heading angle. Concurrently, the rudder is commanded to have zero deflection through the use of the angle of sideslip. Such a combination of commands allows for coordinated turns and the minimization of sideslip. Figure 3.21 illustrates how this autopilot function works. Here, the gains Can be adjusted to increase or decrease the turn rate. The ability of the autopilot to maintain a commanded heading is illustrated in Figure 3.22. The UAV is capable of making large turns, losing altitude at first and then regaining it when the correct heading is acquired. The UAV is able to turn 180° within a radius less than 100 ft, again meeting the requirements of the simulation.

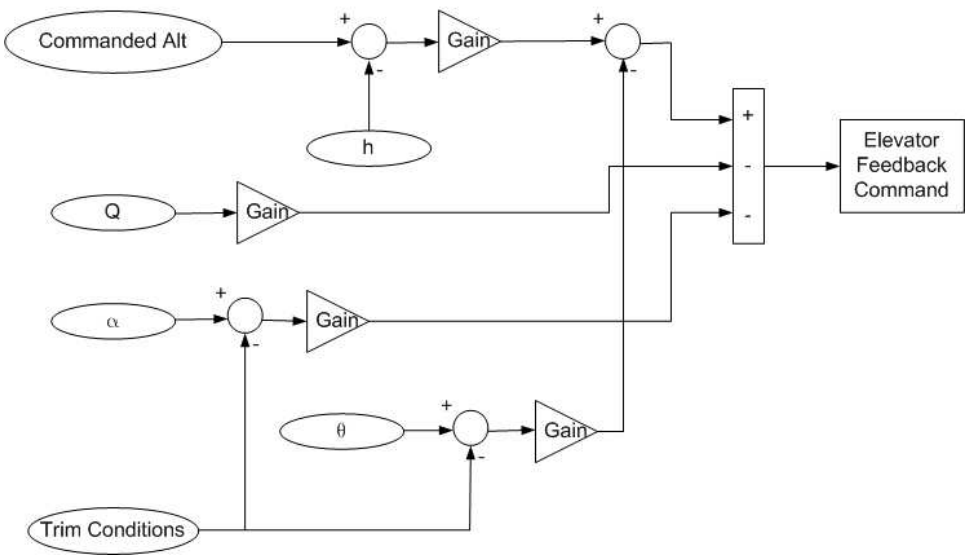


Figure 3.17: Structure of altitude hold feedback loop.

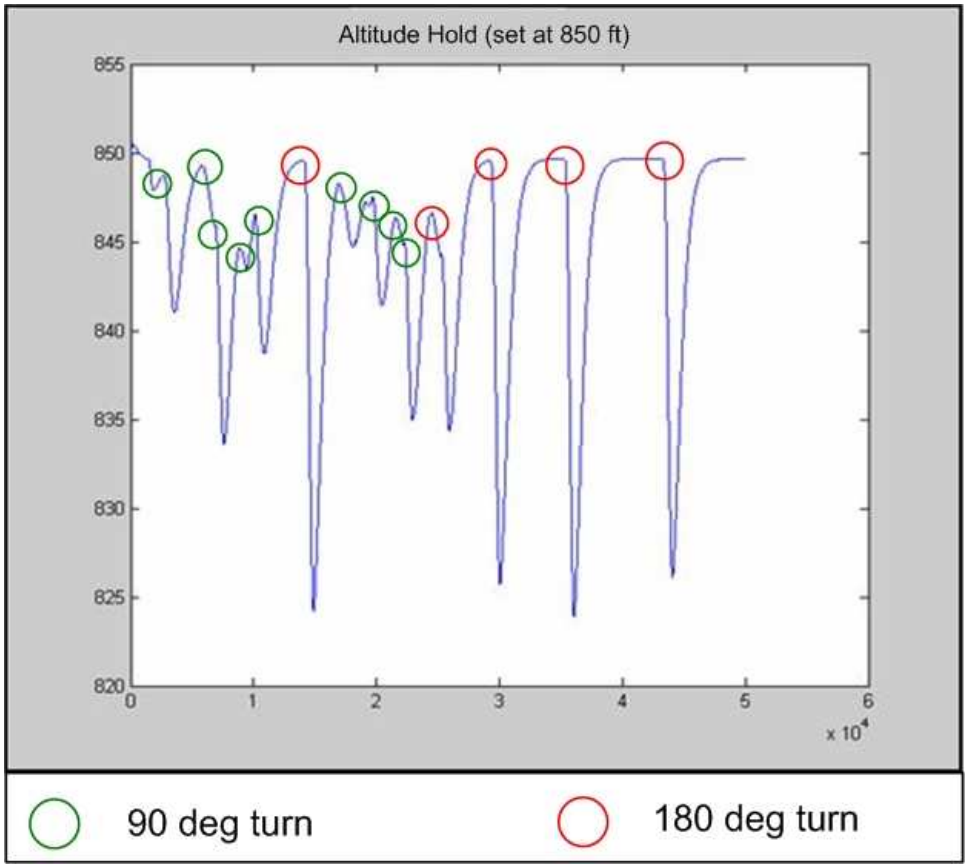


Figure 3.18: UAV altitude hold performance. Example of how well the aircraft model maintains an 850 ft altitude while performing 90° and 180° turns.

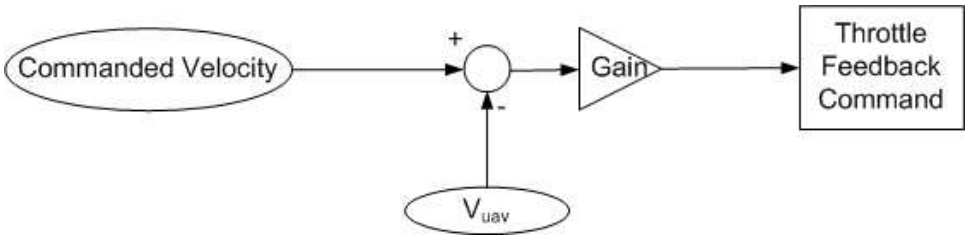


Figure 3.19: Structure of velocity hold routine.

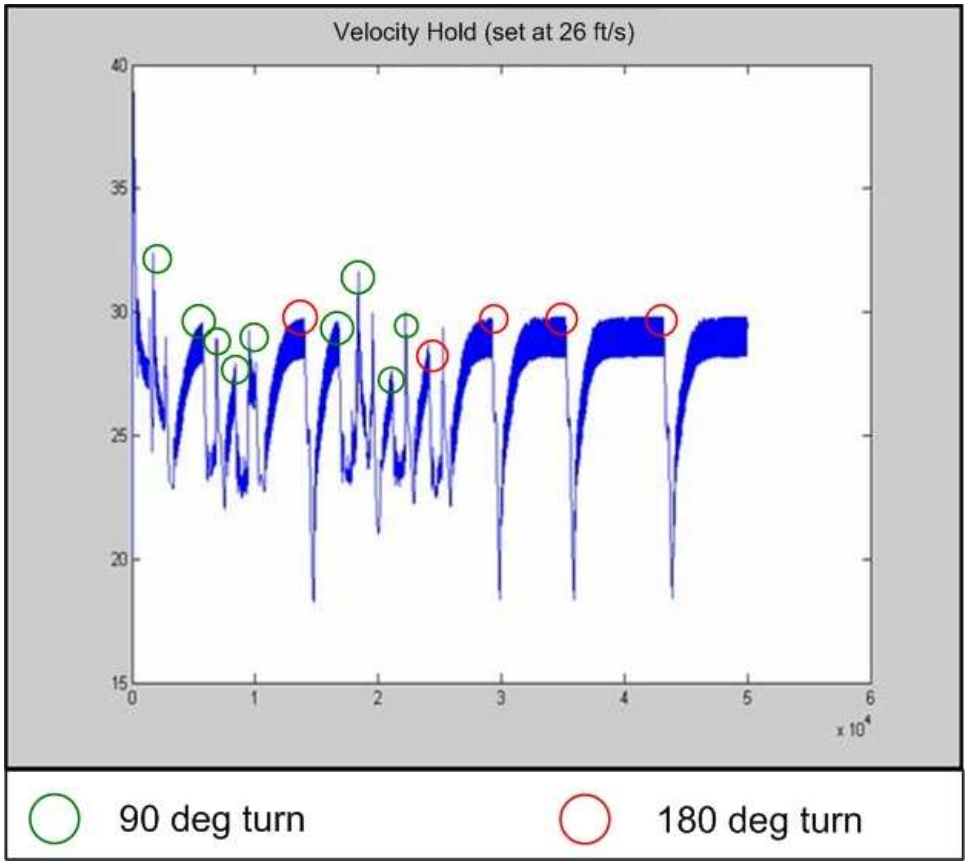


Figure 3.20: UAV velocity hold performance. Example of how well the aircraft model maintains a velocity of 26 $\frac{\text{ft}}{\text{s}}$ while performing 90° and 180° turns.

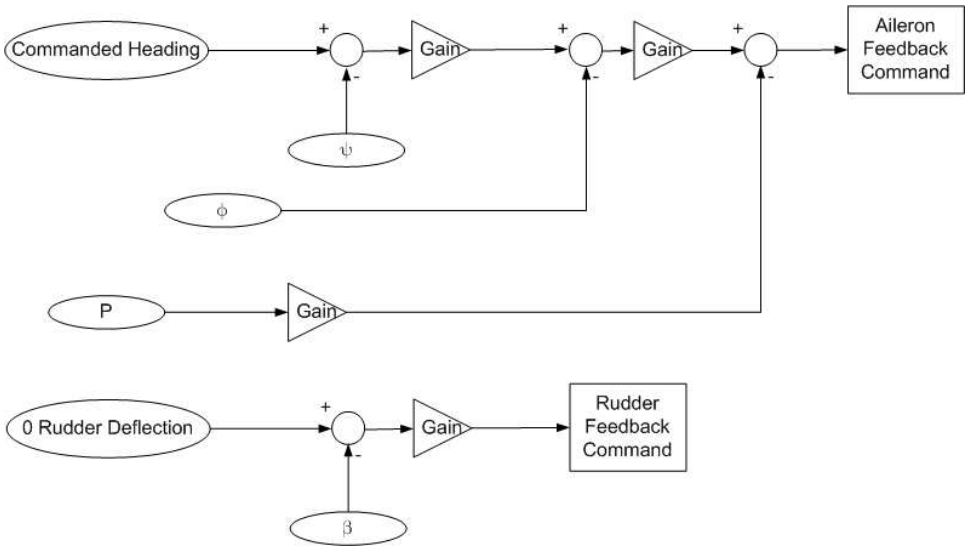


Figure 3.21: Structure of heading hold routine.

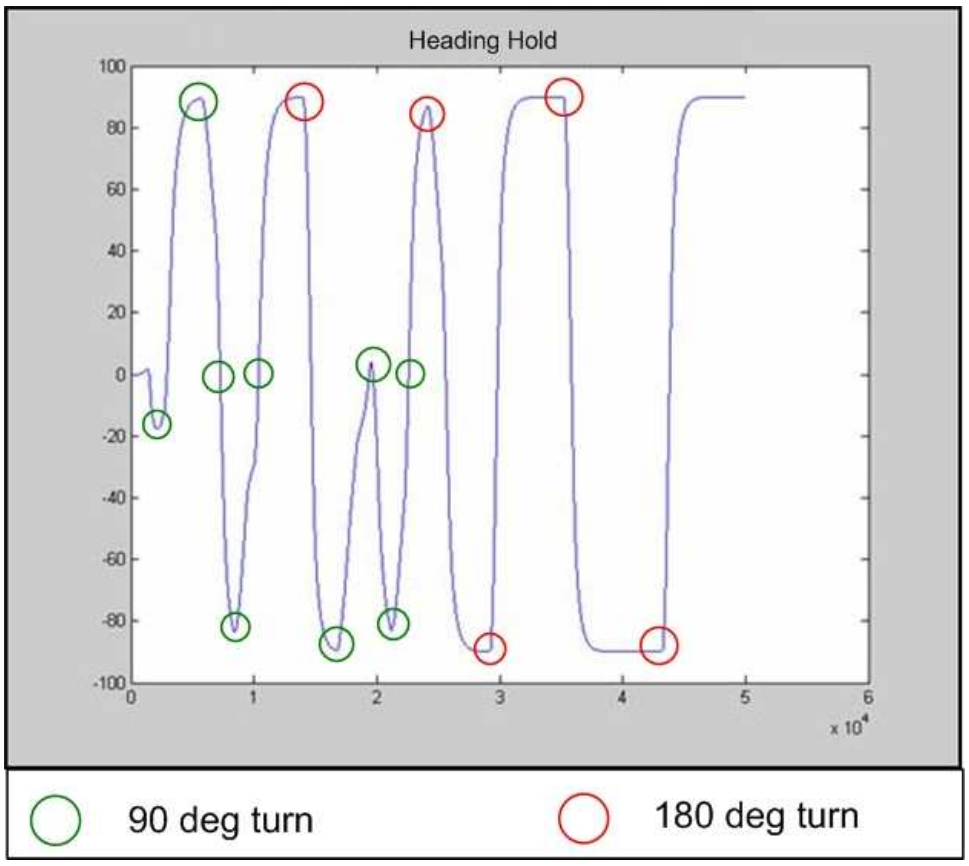


Figure 3.22: UAV heading hold performance. Example of how well the aircraft model maintains a given heading and how quick it is able to respond to a change in heading.

With the UAV dynamics model now introduced, the crux of the thesis research, the sensor model and 3-D navigation algorithm can now be discussed.

3.2.3 Sensor Model. The sensor model is virtually identical to its 2-D counterpart. The only difference is a sphere, with user defined radius, is used to detect the plume, versus a circle. This is a binary detector which has been common practice for computer simulations of odor-based navigation schemes.

3.2.4 Navigation Algorithm. The UAV navigation algorithm consists of four tracking/search schemes: Tracking, Horizontal Search, Backtrack, and Vertical Search. The schemes were developed from a mix of bio-inspiration and ad hoc engineering approaches. The Tracking scheme is based primarily on the "moth-like" STM algorithm and is unique to this thesis effort. The Horizontal Search scheme is directly inspired by the moth's casting behavior and is similar to the searching behavior in the "moth-like" algorithm. Backtrack is a method of relocating the plume, once thought lost, by returning to the vicinity of the last detection. This methodology was adapted from the 2-D robotic simulation found in [12]. The Vertical Search scheme is unique to this thesis effort, developed using ad hoc engineering approaches. Figure 3.23 illustrates how these track/search schemes are intertwined, forming the complete navigation algorithm.

The information used to make decisions in this algorithm is virtually identical to that used in the "moth-like" algorithm. Since the autopilot control system uses altitude (h_{new}), velocity (V_{new}), and heading (θ_{new}) as inputs to drive the dynamics model, these are the variables the navigation algorithm must supply to the autopilot.

3.2.4.1 Tracking. The tracking scheme is based on the same premise as the STM implementation in the "moth-like" algorithm. The UAV's decreased maneuverability and increased velocity drove the need for a longer memory to help with the decision of its location within the plume. The length of the memory, L_{uav}, was initially extended to 10 s, consisting of sensor data collected at 10 Hz (i.e. 100

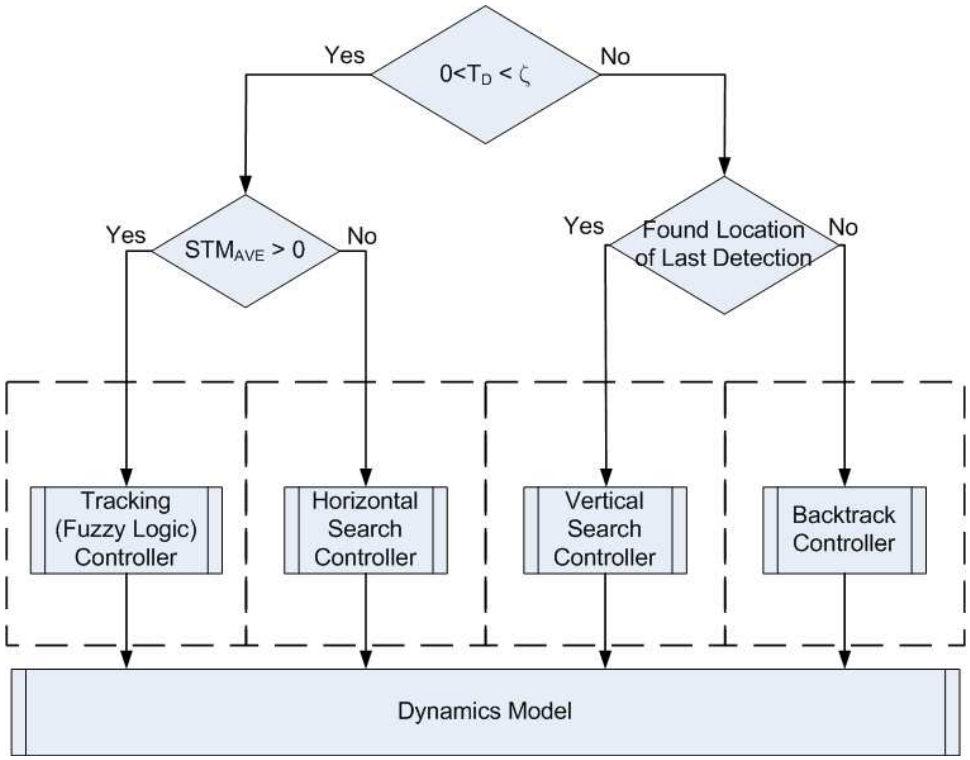


Figure 3.23: Overview of 3D navigation algorithm.

memory locations). Figure 3.24 illustrates this STM. The position of the UAV, relative to the plume, is found by taking the mean of the locations in memory where detections occurred. This is the same method as previously used by the "moth-like" algorithm. Hence, the locations are divided up in a similar fashion:

$$STM_{ave} = \begin{cases} 0 & , \quad \text{Out of plume} \\ 1-40 & , \quad \text{Entering the plume} \\ 30-80 & , \quad \text{In the plume} \\ 60-100 & , \quad \text{Leaving the plume} \end{cases}$$

The UAV uses this tracking algorithm as long as detections exist in the STM. L_{uav} will be one of the parameters varied in the sensitivity analysis presented in Chapter IV. This analysis should bring out the most efficient length of L_{uav}.

Again, due to the ambiguity in the UAV's location relative to the plume, it is pragmatic to use a fuzzy controller to generate the maneuver decision. Figure 3.25

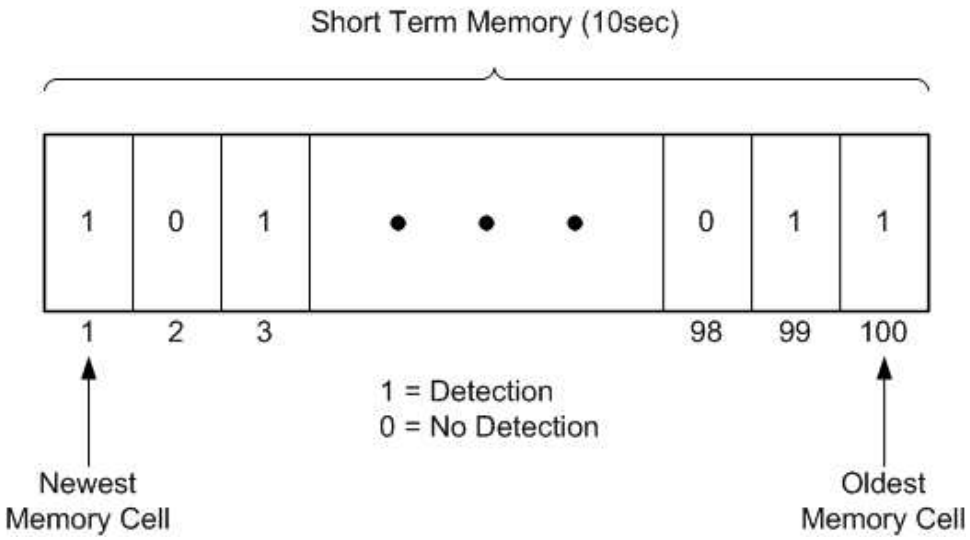


Figure 3.24: UAV short term memory. Illustration of 10 second long STM.

illustrates how the tracking fuzzy controller is designed. Notice the inputs and outputs are identical to the moth tracking controller. The fuzziness of the STM is described by the overlap between the membership functions of the STM fuzzy set, as seen in Figure 3.26

The values used in the fuzzy set for T_Z were increased for the UAV implementation. This increase is due to the UAV capabilities and the overall increase in the size of the plume and area of operation as compared to the moth simulation. The initial layout of the T_Z membership functions is given in Figure 3.27. The values of the membership functions were set by assuming an average velocity and a desired distance to be travelled before executing a counterturn. These parameters are also be varied during the sensitivity analysis in order to determine the optimal setting.

The relative heading input, as well as the new relative heading output fuzzy sets are identical to those of the "moth-like" algorithm and are illustrated by Figure 3.28. The rules governing how the inputs are mapped to outputs are similar to those used in the moth algorithm and are given in Section A.2.1 of Appendix A.

After the UAV's new heading is calculated, it's new velocity is found using the velocity fuzzy controller as depicted in Figure 3.29. The input $H_{UAV_{delt}}$ is used to

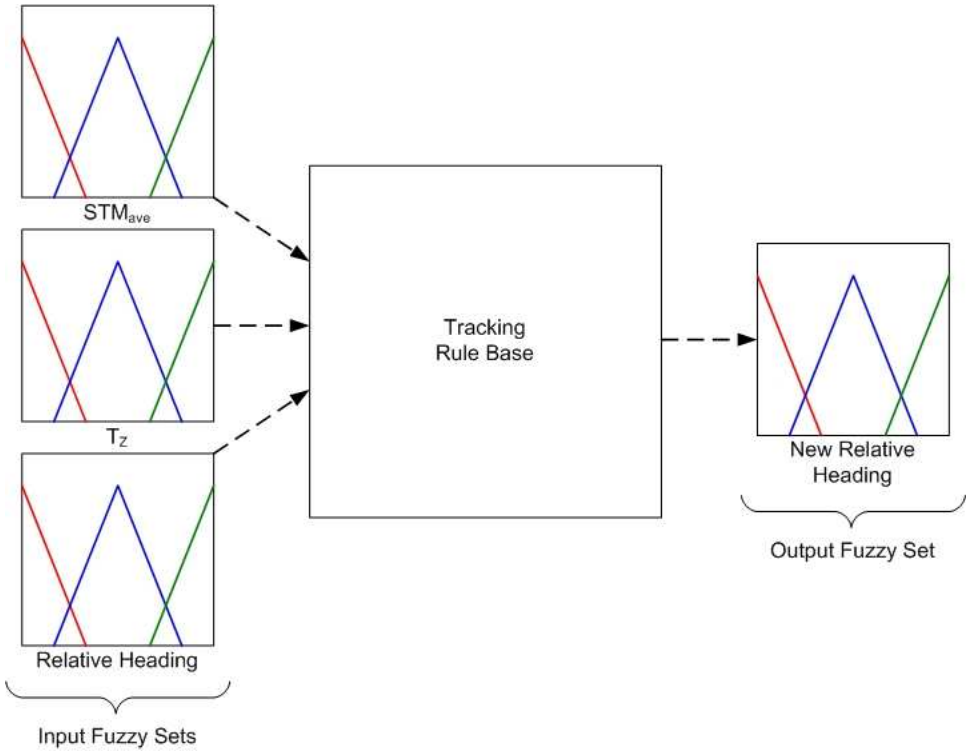


Figure 3.25: Tracking algorithm fuzzy controller. This fuzzy controller contains 3 inputs and 1 output.

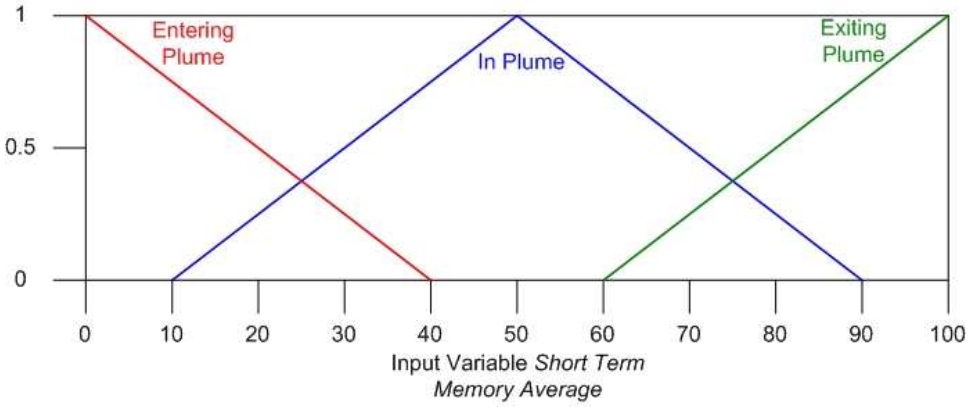


Figure 3.26: STM_{AVE} fuzzy set. The STM is 10 seconds in length with a sampling rate of 10 Hz.

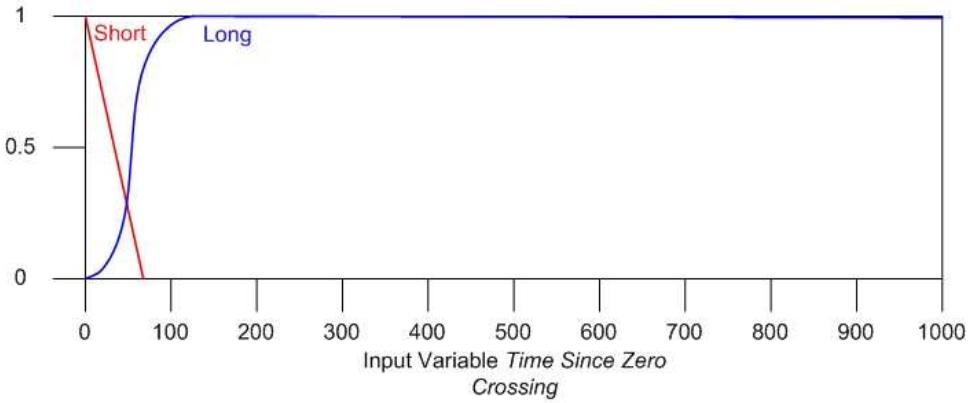


Figure 3.27: T_Z fuzzy set.

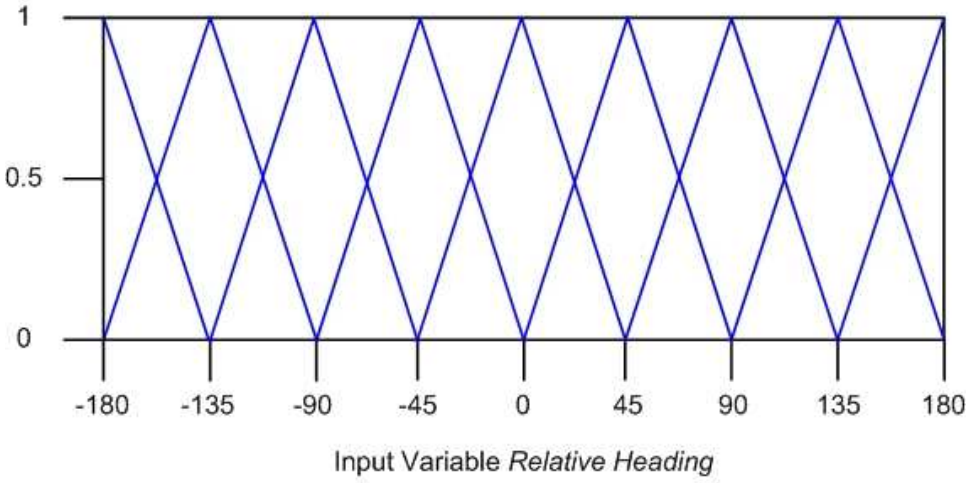


Figure 3.28: Relative heading and new relative heading fuzzy set. Both fuzzy sets are identical and can be represented by the same fuzzy set shown here.

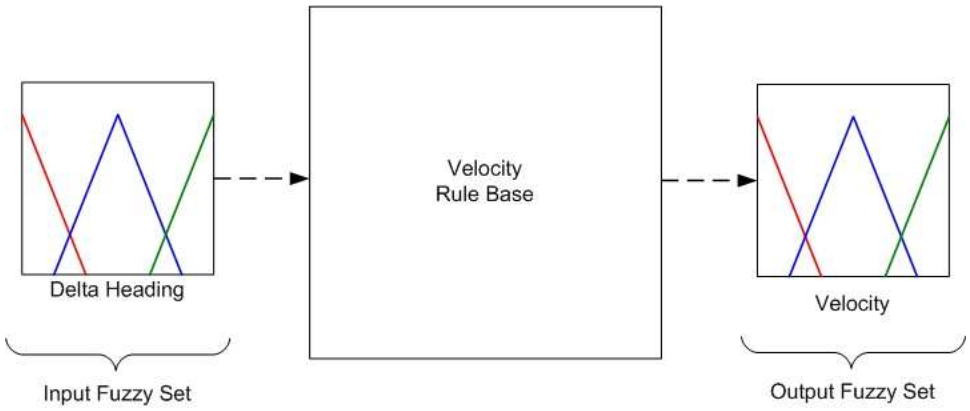


Figure 3.29: Velocity fuzzy controller. Using $H_{UAV_{delt}}$ to dictate $V_{UAV_{new}}$.

determine $V_{UAV_{new}}$, via:

$$H_{UAV_{delt}} = H_{UAV_{new_{rel}}} - H_{UAV_{rel}} \tag{3.37}$$

The associated fuzzy set is the same as the "moth-like" version illustrated in Figure 3.30. The velocity output fuzzy set is similar to the "moth-like" version with the limits of the fuzzy set modified to represent the UAV's capabilities. This is evident in the illustration of the velocity fuzzy set in Figure 3.31. The rules behind this fuzzy controller are given in Section A.2.3 of Appendix A. These rule were developed from the capabilities of the UAV. In order to make a sharp turn, the aircraft has to decrease its velocity. This allows for a tighter turn without pulling an excessive amount of g's.

3.2.4.2 Horizontal Search. A threshold, ζ, is set on the length of time the Horizontal Search routine is allowed to command the UAV while zero detections occur in the STM. When in Horizontal Search mode, the UAV maintains altitude while conducting a casting maneuver perpendicular to the wind. This maneuver is illustrated in Figure 3.32. The UAV continually increases the distance it travels across the wind line by increasing the time between counterturns, T_{ct}, by ΔT_{ct}. These two parameters were both initially set at 20 s and are always set to the same initial value.

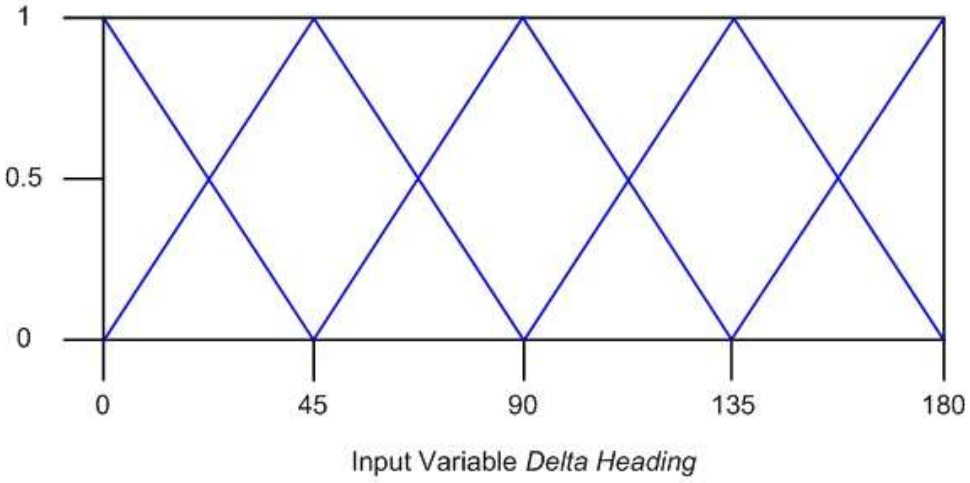


Figure 3.30: $H_{UAV_{delt}}$ fuzzy set.

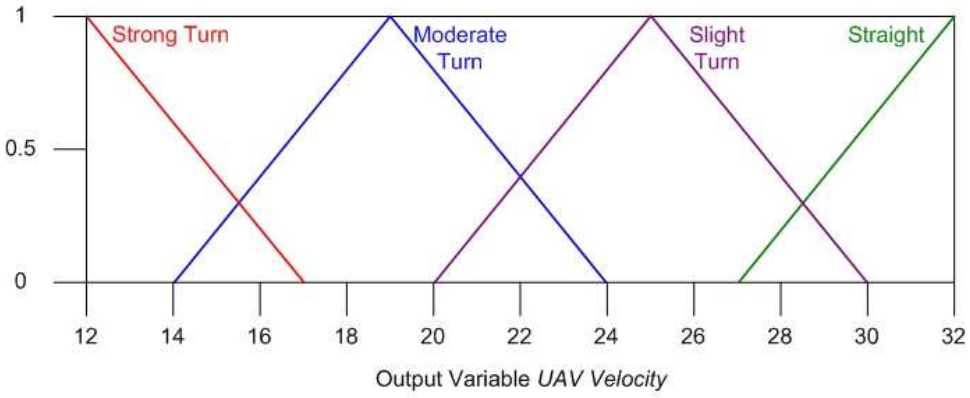


Figure 3.31: V_{UAV} fuzzy set. Executing a sharp turn requires a lower velocity when limiting the amount of stress on the aircraft.

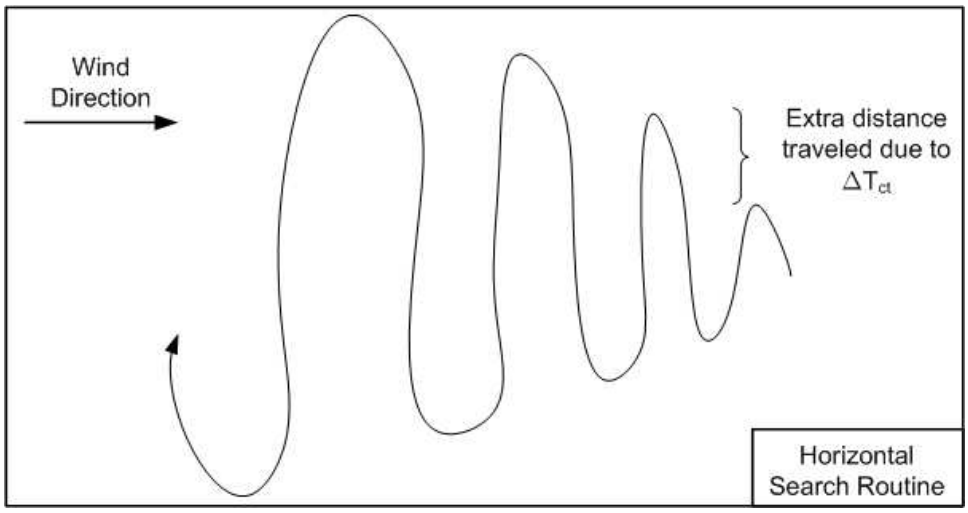


Figure 3.32: Horizontal Search routine. A horizontal casting motion of increasing width occurs when searching the horizontal plane for the plume.

After each counter turn, T_{ct} increases to 40 s, 60 s, 80 s, etc. The parameters ζ, T_{ct}, and ΔT_{ct} will be varied in the simulation study discussed in Chapter IV. The fuzzy controller used to change heading angle, after $T_{ct} + N\Delta T_{ct}$, is identical to the one used in the 2-D simulation. The input and output fuzzy sets are identical and are represented by the same fuzzy set as depicted in Figure 3.28. The associated rules are given in Section A.2.2 of Appendix A. This search routine terminates when the UAV makes a detection or the threshold, ζ, is reached. If a detection is made, the Tracking routine is executed, but if ζ is reached before a detection occurs, the Backtrack routine is executed.

3.2.4.3 Backtrack. This routine is executed when the UAV has lost the plume. Farrell et al. [12] used a method in their 2-D robotics navigation algorithm that guided the robot back to the position where it last made a detection. This method worked quite well in their 2-D experiments, and was used as the basis for the Backtrack routine. Also serving as inspiration for this routine is the observations of MSexta during wind tunnel tests. The moths appear to use visual cues when

Table 3.1: Values used for altitude change, Δh.

Current Altitude (ft)	Δh (ft)
$600 \leq h$	100
$200 \leq h < 600$	50
$50 \leq h < 200$	25
$h < 50$	10

attempting to flying downwind, in hopes of relocating the plume (personal discussions with Dr. M.A. Willis).

The UAV is assumed to have GPS capabilities on board, and has the ability to store the location of the last detection in memory. When Backtrack is executed, the UAV returns to within a certain radius, r_{BT}, of the horizontal location of the last detection. However, the altitude is decreased by Δh, which is varied with altitude. Table 3.1 gives the range of values for Δh. The assumption the UAV flew out of a rising plume is the reasoning behind the development of this routine. This of course assumes a rising plume, which is not always the case. In the simulation study, discussed in Chapter IV, this routine is modified to return the UAV back to the exact location of the last detection, providing insight into potential benefits of either routine when tested against varying plume structures. The Backtrack routine in which the UAV returns to a location below the last detection is illustrated in Figure 3.33. This routine is terminated under two conditions. If a detection is made, the Tracking routine is started. If the UAV reaches the horizontal location of the last detection, the Vertical Search routine begins.

3.2.4.4 Vertical Search. The bulk of the UAV's movement up to this point has taken place in the horizontal plane, with little change in altitude. The assumption is that the mean wind direction does not change drastically over a short period of time (10's of minutes). The plume should still be in the same vicinity as it was during the last detection. Therefore searching the horizontal plane over varying altitudes should result in a detection.

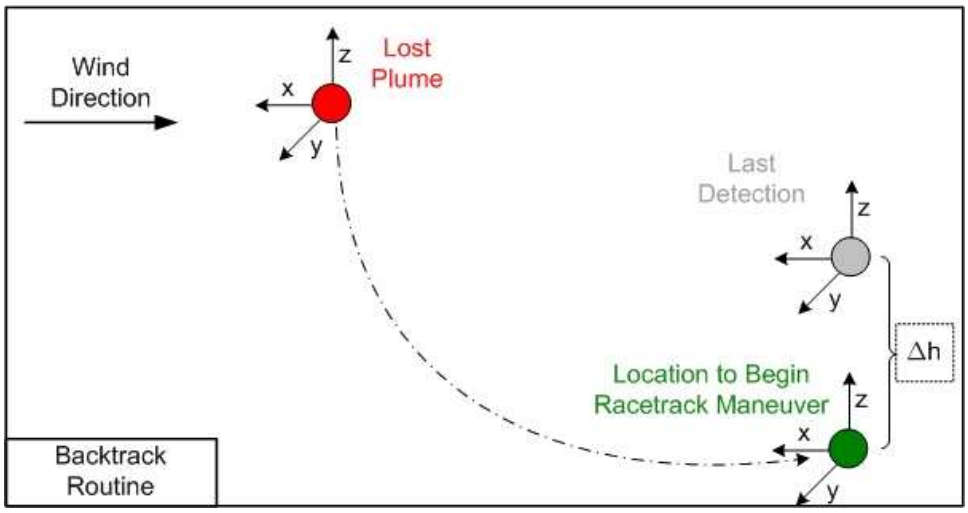


Figure 3.33: Backtrack routine. Once the plume is declared "lost", the UAV travels to Δh below the location of last detection.

The routine begins by travelling in a race track pattern at the altitude dictated by the Backtrack routine (Δh below the altitude of the last detection). Once this pattern is complete, the UAV decreases in altitude by Δh, executing the race track flight profile again. The race track profile is executed 4 more times, except instead of decreasing in altitude the UAV increases its altitude each time by Δh. A detailed illustration of this routine is shown in Figure 3.34.

If at any point a detection is made, this routine is terminated and the Tracking routine begins. If all six race tracks are completed without a detection, the UAV is declared lost and the simulation ends.

3.3 Summary

It should now be apparent that developing 2-D and 3-D simulation environments for the associated odor-based navigation algorithms is not a trivial task. The design of the navigation algorithms for each of these environments required the incorporation both bio-inspired tracking and searching techniques as well as ad hoc engineering approaches. With the navigation algorithms and associated simulation environments thoroughly discussed, the information on the simulation studies performed can be

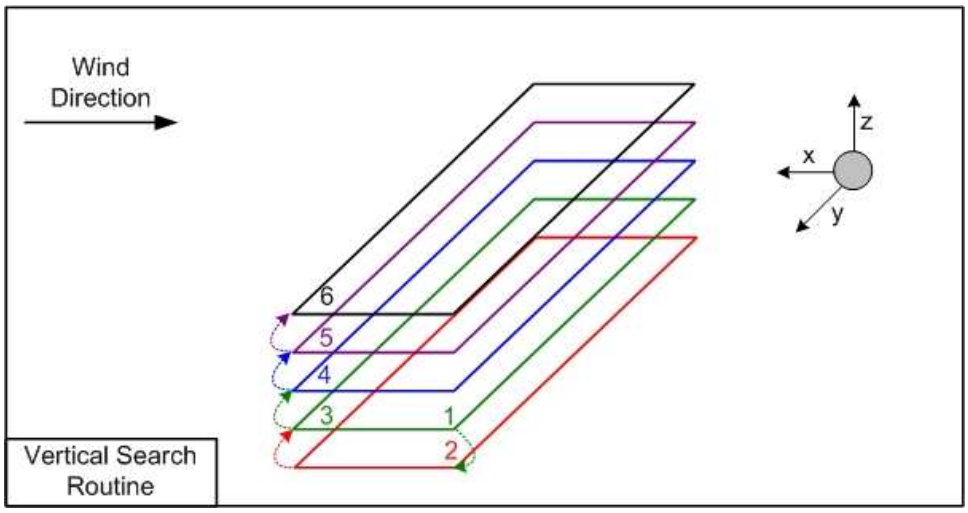


Figure 3.34: Vertical Search routine. Six Race tracks are performed at varying altitudes around the horizontal location of the last detection.

presented. Chapter IV discusses the simulation studies and a brief synopsis of their results.

IV. Simulation Study

Two separate simulation studies were conducted for this thesis: 2-D "moth-like" simulations and 3-D UAV simulations. Many variables within the navigation algorithms are adjustable, with the potential to effect the aircraft's performance. Initial simulation analysis revealed those parameters that were the most influential, and, due to time constraints, only these variables were changed in the final simulations. The 2-D navigation algorithm and simulation were developed as a stepping stone to the development of the 3-D algorithm/simulation. This led to dedicating more time and effort on the 3-D simulation study than its 2-D counterpart. Therefore, a wider range of variables were tested in the 3-D study in comparison to the 2-D study. Environmental variables were also changed. Testing the navigation algorithms against different environments allows more insight on the effectiveness and robustness of the algorithms. The two environmental variables modified were the plume structure and sensor size. As discussed in Chapter III, sensor size is actually a reflection of plume density rather than the sensor's sensitivity, which is why it is labeled as an environmental variable.

4.1 Design of Experiments

This section discusses the reasoning behind the variation of parameters chosen for both the 2-D and 3-D simulation studies. The parameters varied can be associated with either the plume generation or navigation algorithm.

4.1.1 2-D Simulation Design. The different plume parameters that can be altered, changing its structure, are:

1. Mean wind velocity
2. Mean wind direction
3. Frequency of wind meandering
4. **Density of plume (i.e., sensor size)**
5. **Initial plume structure (i.e., change initial meandering direction)**

6. Location of source

The items highlighted in boldface (numbers 4 and 5) were chosen as parameters to be altered during this simulation study. In addition, many parameters associated with the navigation algorithm could have been varied. However, due to time constraints only parameters associated with the plume were altered. A list showing all combination of the varying parameters for this simulation study are given in Table 4.1. Plume A and B have different initial plume structures, otherwise they are identical.

Table 4.1: 2-D simulation scenarios.

Plume meandering	Sensor Size (cm)
A	0.3
A	0.4
A	0.5
A	0.6
A	0.7
B	0.3
B	0.4
B	0.5
B	0.6
B	0.7

A Monte Carlo analysis, consisting of 100 runs, was conducted for each scenario. This number of runs was chosen due to time constraints. One scenario of 100 runs took approximately 1.5 hours to complete. Although this is not an exorbitant amount of time, this simulation study was completed after the 3-D study and time was of the essence.

The statistics of interest, from the Monte Carlo analysis, are:

1. Number of successful runs (i.e., the moth reaches the source)
2. The mean and standard deviation of the time it takes the successful runs to reach the source

It is expected that robustness (the ability of a navigation algorithm to successfully navigate varying plume structures) will be sacrificed for performance (the ability of the navigation algorithm to quickly reach the source).

4.1.2 3-D Simulation Design. As was the case in the 2-D simulation design, parameters of both the plume and navigation algorithm can be varied in testing the design of 3-D navigation algorithm. The plume parameters that can be varied are identical to those discussed in Section 4.1.1. However, the plume parameters altered are:

1. **Frequency of wind meandering**
2. **Density of plume (i.e. sensor size)**
3. **Location of source**

Four plume structures were used, each tested at varying densities:

1. B - Rising, slowly meandering plume
2. C - Rising, quickly meandering plume
3. D - Level, slowly meandering plume
4. E - Level, quickly meandering plume

More detail on these plumes is given in Section 4.4.

Since the 3-D navigation algorithm is of greater importance to this thesis effort, more time was allowed for the simulation study. This allowed navigation parameters to be varied in addition to plume parameters. The parameters of the navigation algorithm which could provide increased performance or robustness are:

1. Length of STM
2. Membership functions within the STM fuzzy set
3. Membership functions within the T_Z fuzzy set

4. ζ

5. $\mathbf{\Delta T_{ct}}$

6. Variations on the Vertical Search routine

7. **Variations on the Backtrack routine**

Time constraints only allowed for a small subset of parameters to be varied in the 3-D simulation. The parameters chosen are highlighted in boldface. The parameters ζ and ΔT_{ct} were chosen because they have a large effect on how the UAV conducts its horizontal maneuvers. The Backtrack routine was chosen because it gives the UAV the ability of relocating the plume once it is lost. Optimizing this routine would benefit the aircraft greatly.

Two sets of simulations were conducted: Initial and Final. Only the parameters ζ and ΔT_{ct} were altered (along with the four varying plume structures, single density) in the Initial study. The scenarios tested are given in Table 4.2.

Table 4.2: Scenarios tested for 3-D Initial study (tested against plumes B-E).

ζ (s)	ΔT_{ct} (s)
240	20
300	20
180	20
420	20
240	10
240	30
180	10

The purpose was to attain a subset of the variations of parameters tested. This subset of better performing parameters is used in the Final simulation study. The Final study uses these parameters along with a variation of the Backtrack routine in testing against the same 4 plume structures with varying densities. More detailed information for both the Initial and Final simulation is given in Section 4.4.

A Monte Carlo analysis, consisting of 10 runs, was conducted for each scenario. This number of runs was chosen due to time constraints. One scenario of 10 runs took approximately 5.5 hours to complete and was the main reason for having to limit the parameters varied.

The statistics of interest, from the Monte Carlo analysis, are:

1. Number of successful runs (i.e., the UAV reaches the source)
2. The mean and standard deviation of time it takes the successful runs to reach the source

It is expected that robustness (the ability of a navigation algorithm to successfully navigate varying plume structures) will be sacrificed for performance (the ability of the navigation algorithm to quickly reach the source).

4.2 Orientation to the Data

In order to fully comprehend the figures depicting the plumes and moth/UAV trajectories, a short orientation to these figures is needed.

4.2.1 2-D Orientation. Figure 4.1 is an example of a 2-D plume. The source of the plume is always located at (0,0.5) (x-direction,y-direction), as indicated by the red circle. The mean wind direction is always in the left to right direction, as denoted by the arrow. Figure 4.2 illustrates data from an actual 2-D simulation, showing two plumes. The green plume is the initial plume (at time = 0) and the red plume is the final plume (at the time the simulation ends). This figure also shows the trajectory of the moth in blue. There is a distinct difference in the moth's behavior depending on its navigation mode, tracking (NAV1) or casting (NAV2), which are highlighted in the figure.

4.2.2 3-D Orientation. There are two main types of plumes in the 3-D simulation study, one that starts from the ground and rises (Figure 4.3), and one that starts at an altitude of 500 ft and decreases slightly downwind (Figure 4.4). Plumes

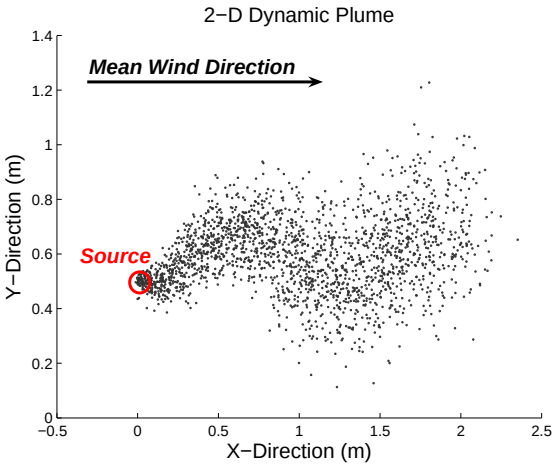


Figure 4.1: 2-D Dynamic Plume Example.

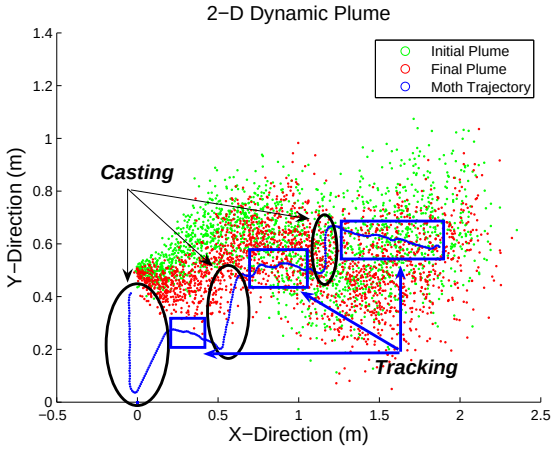


Figure 4.2: 2-D Simulation Example.

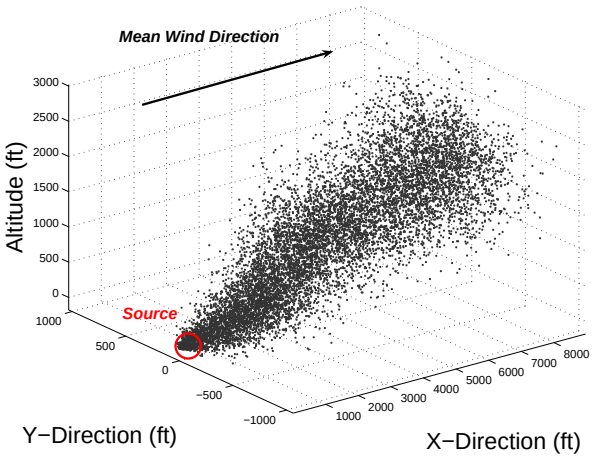
B and C of the 3-D simulations are of the first type mentioned. Plumes D and E are of the second type. The mean wind direction is in the x-y plane, flowing from left to right. The source for plumes B and C are (0.5, 0, 0.5) while for plumes D and E it is located at (0.5, 0, 500).

The UAV trajectory is more complicated than that of the 2-D simulation, an example is illustrated in Figure 4.5. Here, one can see the trajectory is broken into green, blue, black and red sections. The colors represent the 4 navigation routines currently being used:

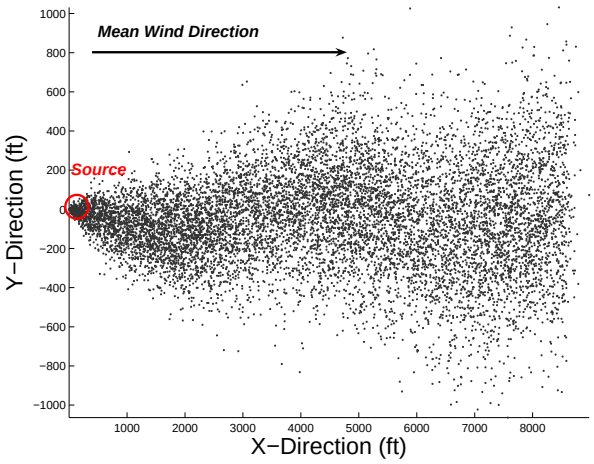
1. *Green:* Tracking via STM
2. *Blue:* Horizontal Search
3. *Black:* Backtrack
4. *Red:* Vertical Search

Figure 4.6 illustrates a close-up section of the UAV's trajectory from Figure 4.5. The black points indicated by the number 1 and letter A are the locations where the UAV transitions from the Horizontal Search routine to the Backtrack routine. The UAV then transitions from the Backtrack routine to the Vertical Search routine, indicated by the number 2 and letter B. After a short period of time in the Vertical Search routine, the UAV detects the plume and returns to the Tracking routine, represented by number 3 and letter C. As the STM_{ave} decreases to zero, the UAV switches to the Horizontal Search routine, indicated by number 4 and letter D.

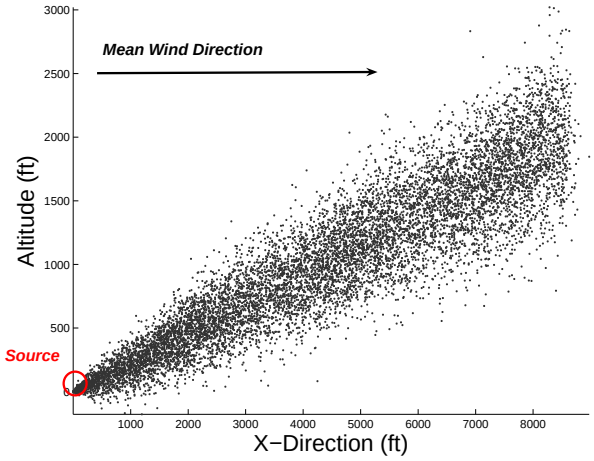
On the other hand, Figure 4.7 depicts the UAV completing the Vertical Search routine, ending the simulation with a failure to find the source. The six levels of the Vertical Search routine are labeled in the order they occurred. It is obvious the pattern does not exactly match the description given in Chapter III. However, the fundamentals of what the pattern is supposed to achieve (scanning horizontally over various altitudes) is present.



(a) 3-D view

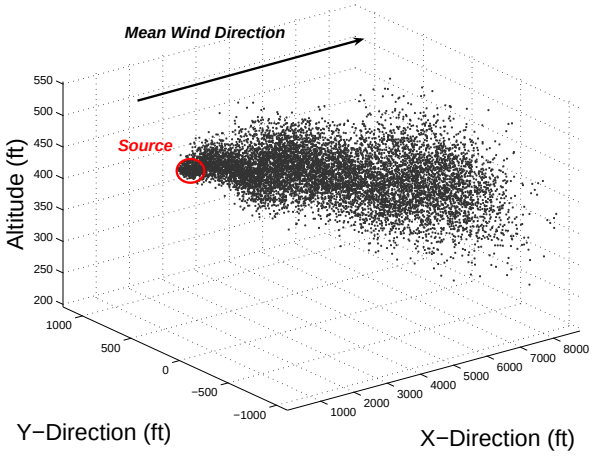


(b) Horizontal plane

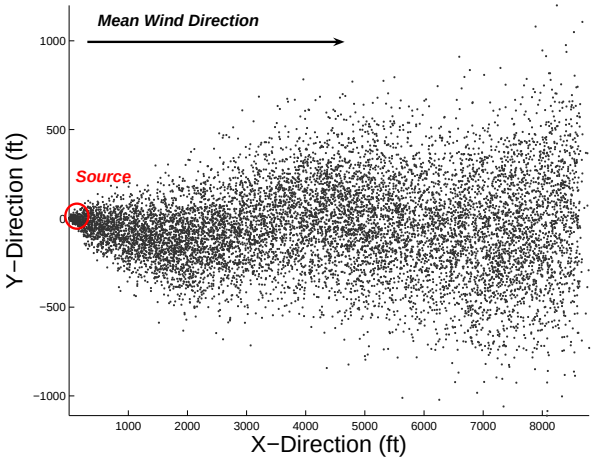


(c) Vertical plane

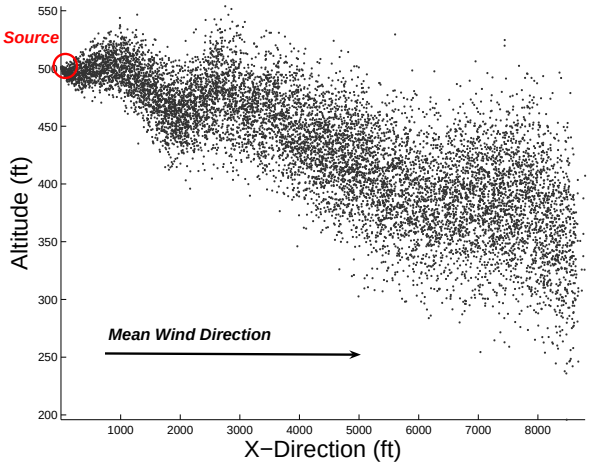
Figure 4.3: 3-D Dynamic Plume Example (Rising Plume). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

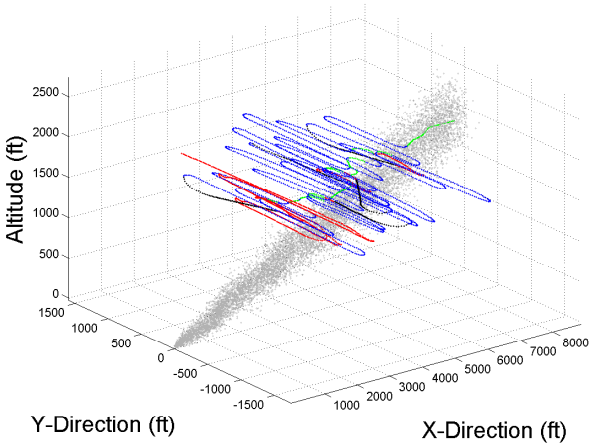


(b) Horizontal plane

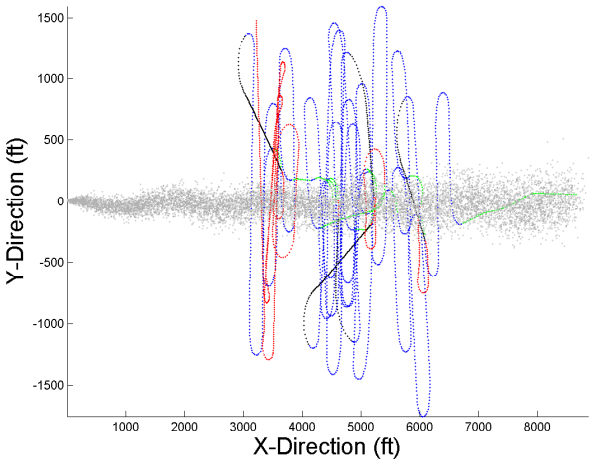


(c) Vertical plane

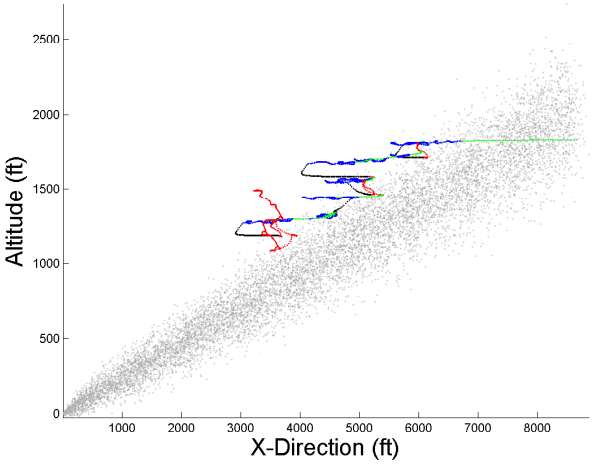
Figure 4.4: 3-D Dynamic Plume Example (Rising Plume). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

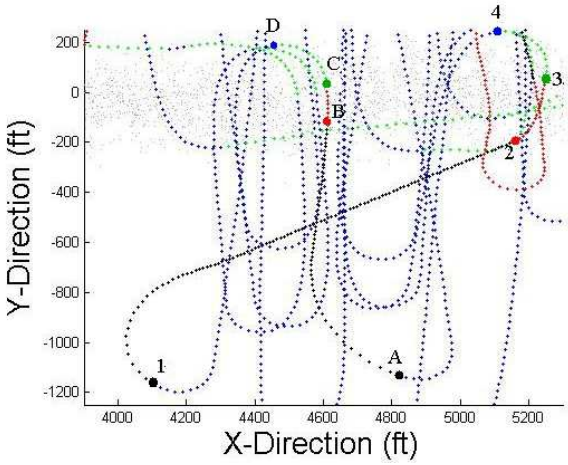


(b) Horizontal plane

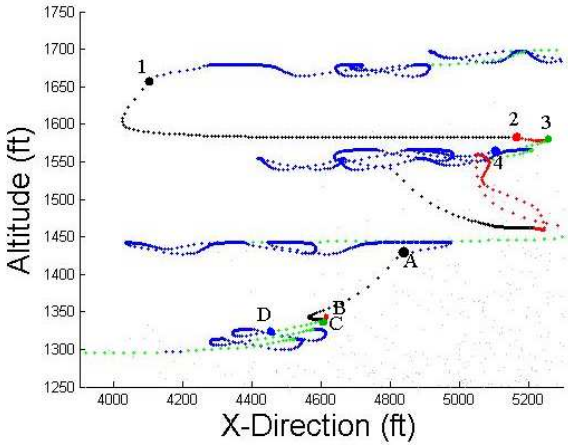


(c) Vertical plane

Figure 4.5: 3-D Dynamic Plume Example (Rising Plume). (a) 3-D view (b) Horizontal plane (c) Vertical plane

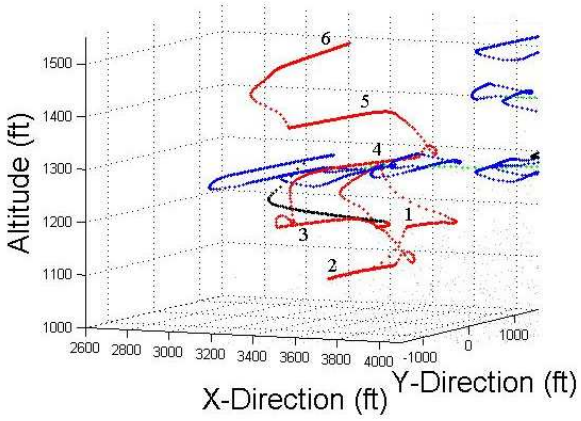


(a) Horizontal plane

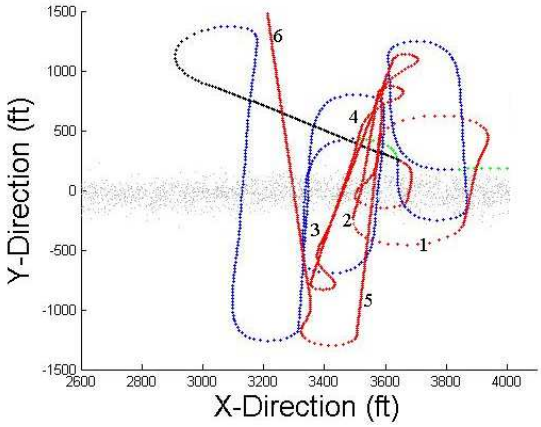


(b) Vertical plane

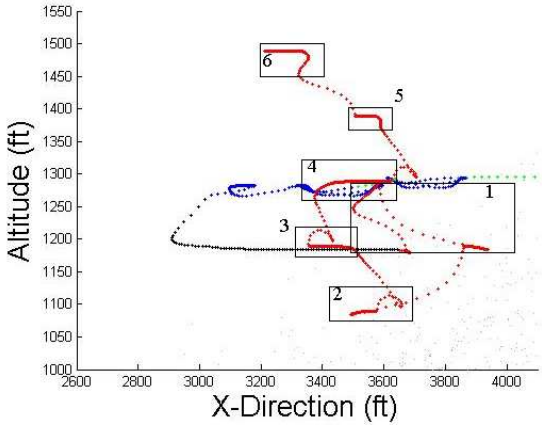
Figure 4.6: Magnified 3-D Trajectory (Successful Transitions Among All Navigation Routines). (a) Horizontal plane (b) Vertical plane



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.7: Magnified 3-D Trajectory (Complete Vertical Search Routine). (a) 3-D view (b) Horizontal plane (c) Vertical plane

4.3 2-D Simulation Study

The purpose of this study was to mimic the results for laboratory wind tunnel experiments of the male M.Sexta navigating a female pheromone plume to its source. In laboratory tests, MSexta boasted an average success rate of up to 70% [5] with a time to source of 10 to 12 s (M.A. Willis, personal communication). This simulation study attempted to match the wind tunnel results, as well as maintain the "moth-like" flight profile. A Monte Carlo analysis consisting of 100 runs was performed on each scenario (different combinations of variable values). Every run had the possibility of terminating in two ways:

1. If the moth travels to within 10 cm of the source in the 20 s timeframe, the run is deemed a success.
2. If the simulation times out before the moth finds the source location, the run is categorized as a failure.

This simulation data was analyzed for both the number of successful runs and statistics on the time needed to reach the source during each successful run.

The baseline navigation algorithm discussed in Chapter III was tested using five sensor sizes against two plume structures (plumes A and B). The five sensor sizes used were: 0.3 cm, 0.4 cm, 0.5 cm, 0.6 cm, and 0.7 cm. These sizes were chosen after preliminary simulation analysis indicated an acceptable range of sizes. If the sensor is too large, the moth will detect the plume even if it is well outside the plume's boundary. This will cause inappropriate maneuver decisions to be made, resulting in repeated failures in locating the source. Also, if the sensor is too small, it will never detect the plume, resulting in a failure to locate the source.

Although plumes A and B are not exact replications of the true plumes in the wind tunnel experiments, they exhibit key realistic characteristics as discussed in Chapter III. Figures 4.8 and 4.9 illustrate plumes A and B, respectively. These plumes are the same, however, they begin meandering in opposite directions. The

plumes stay within the 1 m × 2 m area, equivalent to the size of the wind tunnel in the laboratory experiments mentioned in Chapter II.

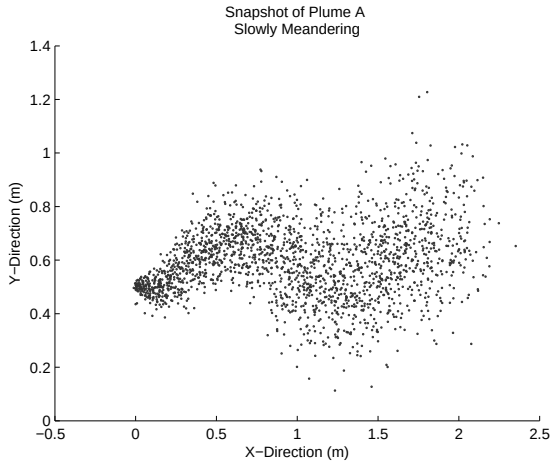


Figure 4.8: 2-D Simulation: Plume A. Initial plume for all simulations tested against plume A.

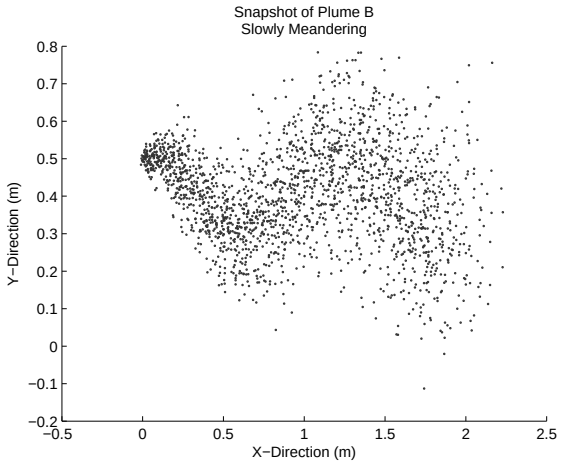


Figure 4.9: 2-D Simulation: Plume B. Initial plume for all simulations tested against plume B.

Tables 4.3 and 4.4 show the results from the Monte Carlo simulations for plumes A and B, respectively. Both plumes generated similar results, including the increasing success of the moth as the sensor size increases. Also, along with its associated standard deviation, the mean time to locate the source decreased. These facts are more easily indicated by a graphical representation of the tabular data. Figures 4.10

and 4.11 illustrate the number of successful runs, as well as the statistics behind those runs for plumes A and B, respectively.

Table 4.3: 2-D simulation results tested against plume A with various sensor sizes (100 runs).

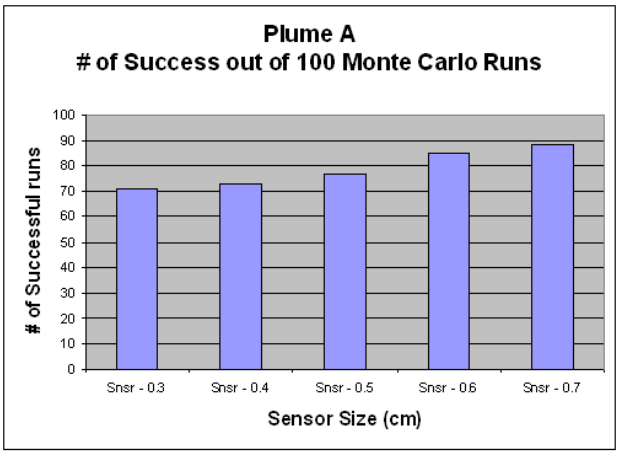
Sensor Size (cm)	Successes (out of 100 runs)	Ave Time of Successes (s)	Stdev
0.3	71	7.31	3.81
0.4	73	5.01	2.65
0.5	77	3.90	2.57
0.6	85	3.07	1.97
0.7	88	2.67	1.78

Table 4.4: 2-D simulation results tested against plume B with various sensor sizes (100 runs).

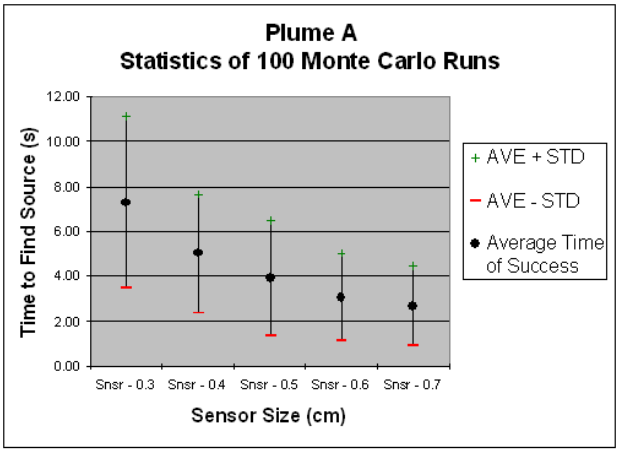
Sensor Size (cm)	Successes (out of 100 runs)	Ave Time of Successes (s)	Stdev
0.3	64	9.09	3.86
0.4	63	7.36	3.54
0.5	74	5.92	3.75
0.6	79	4.93	3.45
0.7	80	4.19	2.89

The larger the sensor (within reason) the greater the chance of detection (i.e., increasing the density of the plume). This corresponds to a more direct upwind flight path, and results in locating the source in less time. Figures 4.12 and 4.13 illustrate successful runs incorporating the 0.7 cm sensor for plumes A and B, respectively. These runs had short location times, 1.88 s and 2.64 s, making their flight path less moth-like. This is standard for all short location times, regardless of sensor size. Therefore, these short times and straight profiles were not representative of the wind tunnel experiments. However, they do suggest an effective navigation algorithm. Alternatively, these highly successful runs might be due to an unrealistic plume density.

The scenarios that exhibited the most "moth-like" trajectories and time-to-source statistics were less successful and incorporated the smaller sensors (i.e., less

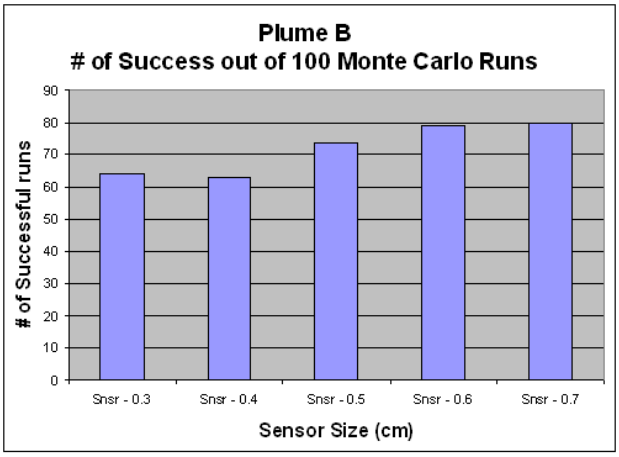


(a) Number of successful runs out of 100

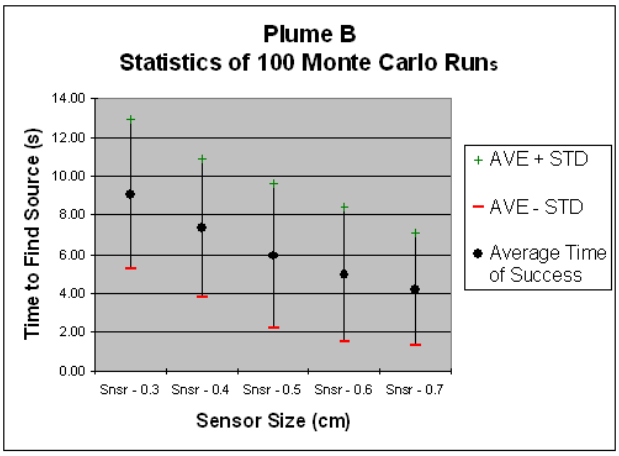


(b) Statistics of successful runs

Figure 4.10: 2-D, plume A simulation results. (a) Number of successful runs out of 100. (b) Statistics of successful runs.



(a) Number of successful runs out of 100



(b) Statistics of successful runs

Figure 4.11: 2-D, plume B simulation results. (a) Number of successful runs out of 100. (b) Statistics of successful runs.

dense plumes). Although these small-sensor scenarios still produced quick location times, they were not as prevalent as those with larger sensors. Figures 4.14 and 4.15 illustrate successful runs with scenarios incorporating the 0.3 cm sensor for both plumes A and B, respectfully. These runs had location times of 11.80 s and 8.51 s and fell in the range of the wind tunnel tests. With the longer location time, comes a more indirect route to the source, increasing the flight time and making the flight path more representative of M.Sexta.

Regardless of sensor size, when a simulation was unsuccessful, the moth typically flew past the source and never regained contact with the plume. This was due to the casting portion of the navigation algorithm having inadequate capabilities. When lost, the algorithm should have commanded the moth to travel further downwind, in an attempt to relocate the plume. Figures 4.16 and 4.17 show unsuccessful trajectories of a scenario incorporating a 0.3 cm sensor navigating plume A and a scenario with a 0.7 cm sensor navigating plume B, respectively. The majority of the simulation time was expended while the moth was casting, searching for the lost plume.

The 2-D simulations incorporating the larger sensors had similar success rates in locating the source as in the wind tunnel experiments. However, these scenarios had a much quicker time-to-source than the wind tunnel data. The scenarios with the smaller sensors produced time-to-source statistics equivalent to that of the wind tunnel experiments, but had approximately a 10% lower success rate. When the time-to-source for the simulations closely matched the wind tunnel results, the profiles became more "moth-like". Further analysis of the results and the conclusions drawn are discussed in Chapter V.

4.4 3-D Simulation Study

The focus of this study was to optimize the 3-D navigation algorithm, increasing the success rate for locating the plume's source and decreasing the time to do so. Two groups of simulations were performed, with the second group using the results found

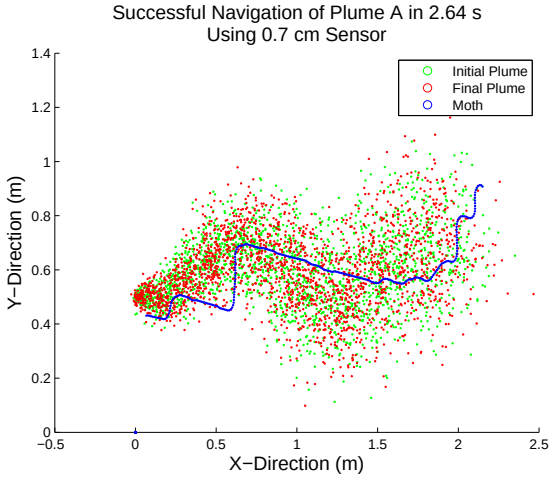


Figure 4.12: 2-D simulation: successful navigation of plume A. This simulation used a sensor size of 0.7 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth locates the source in 2.64 s.

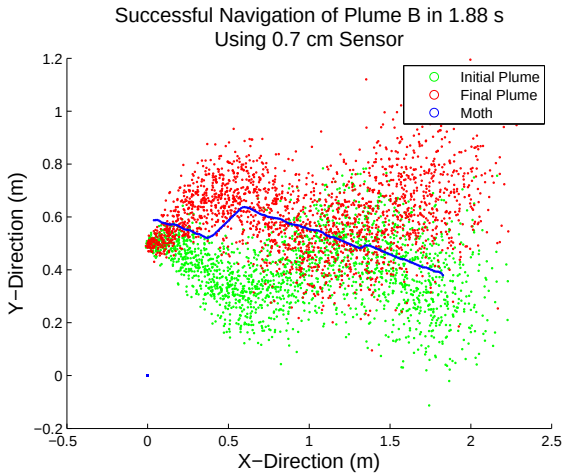


Figure 4.13: 2-D simulation: successful navigation of plume B. This simulation used a sensor size of 0.7 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth locates the source in 1.88 s.

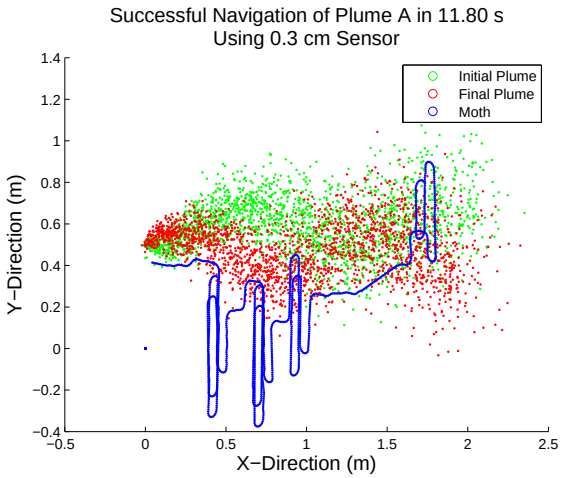


Figure 4.14: 2-D simulation: successful navigation of plume A. This simulation used a sensor size of 0.3 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth locates the source in 11.80 s.

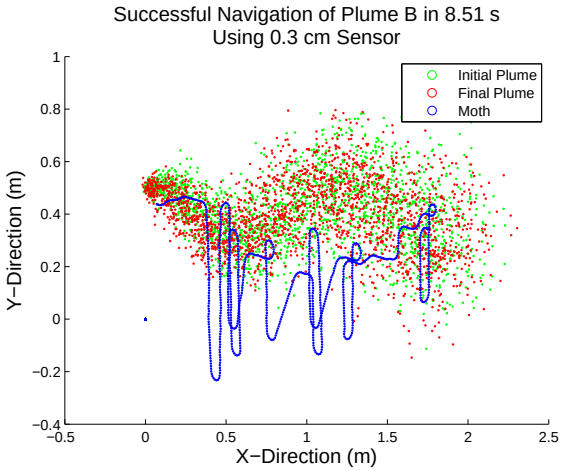


Figure 4.15: 2-D simulation: successful navigation of plume B. This simulation used a sensor size of 0.3 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth locates the source in 8.51 s.

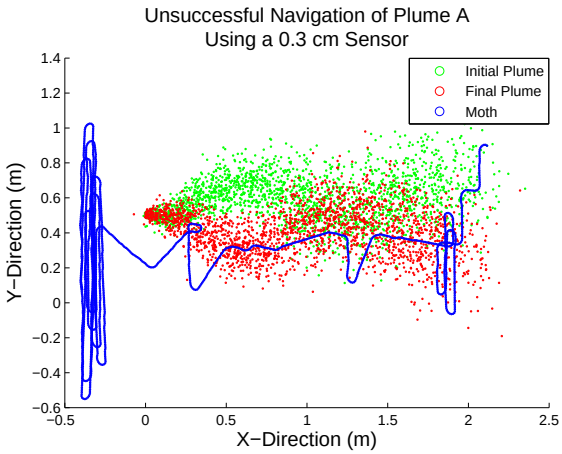


Figure 4.16: 2-D simulation: unsuccessful navigation of plume A. This simulation used a sensor size of 0.3 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth loses contact with the plume and is unable to locate it again.

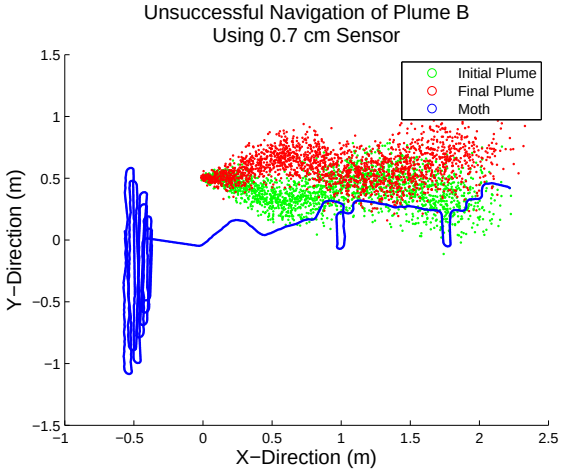


Figure 4.17: 2-D simulation: unsuccessful navigation of plume B. This simulation used a sensor size of 0.3 cm. The red plume is the structure of the initial plume and the green is the structure of the final plume. Here, the moth loses contact with the plume and is unable to locate it again.

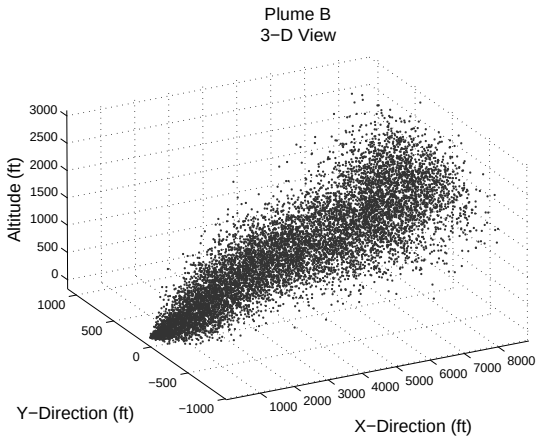
from the first. A Monte Carlo analysis consisting of 10 runs was performed on each scenario. Each run had the possibility of terminating in one of 4 ways:

1. Within the 7000 s simulation timeframe, the UAV travels to within 100 ft of the source, representing the only way for a success to occur.
2. The UAV completes the Vertical Search routine without detecting the plume (failure).
3. The UAV crashes into the ground (failure).
4. The simulation times out before the UAV locates the source (failure).

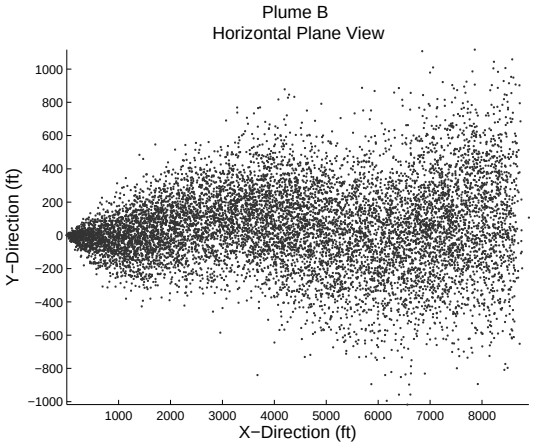
The simulation data was analyzed based on the number of successful runs and the statistics on the length of time it took to reach the source for each of the successful runs.

Both studies were tested against the same four plumes (B through E). The source for plumes B and C was located on the ground (i.e., altitude of 0 ft). These plumes rise quickly in the vertical plane while meandering in the horizontal plane, with plume B meandering slowly while plume C meanders more quickly. Plumes B and C are illustrated in Figures 4.18 and 4.19, respectively. The source for plumes D and E was located at an altitude of 500 ft. These plumes slowly decrease in the vertical plane while meandering in the horizontal plane, with plume D meandering slowly while plume E meanders more quickly. Plumes D and E are illustrated in Figures 4.20 and 4.21, respectively.

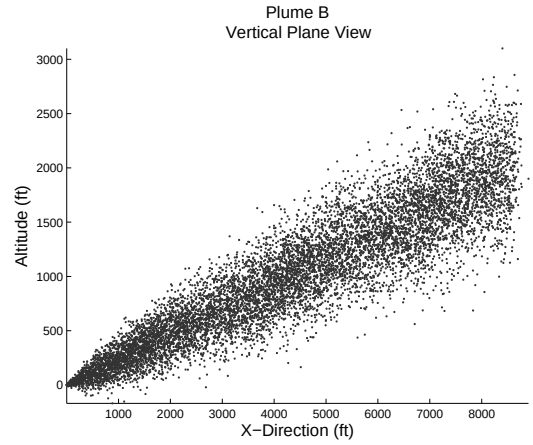
4.4.1 Initial Study. Two variables, which were easily changed and that have significant impact on the UAV's performance, are the threshold for the time since detection, ζ, and the time the horizontal casting is increased, ΔT_{ct}. The threshold, ζ, controls how long the UAV stays in the Horizontal Search routine before switching to the Backtrack routine. Changing ΔT_{ct} allows for narrower or wider search patterns within the Horizontal Search routine. Using these variables, seven scenarios were generated for testing against the four simulated plumes. Table 4.5 gives the values of



(a) 3-D view

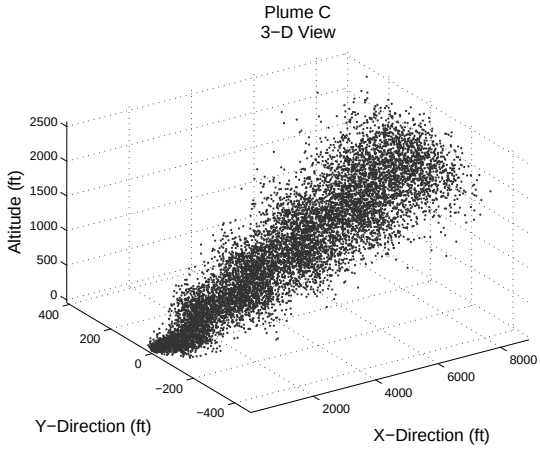


(b) Horizontal plane

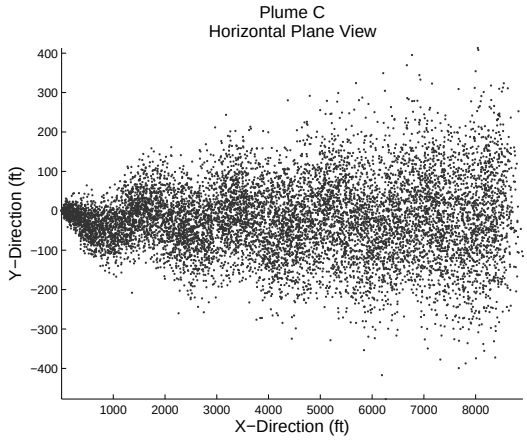


(c) Vertical plane

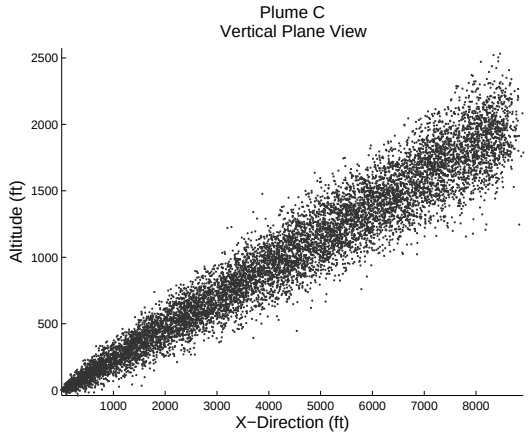
Figure 4.18: 3-D, plume B simulation results. (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

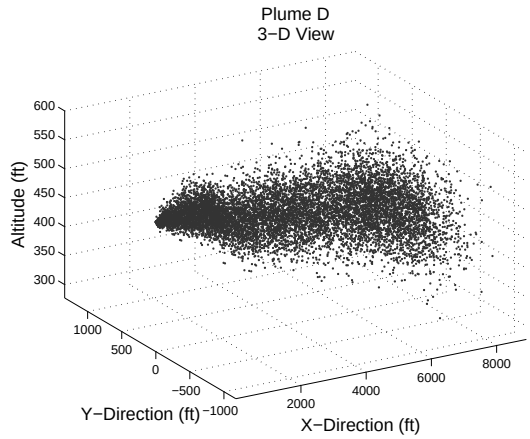


(b) Horizontal plane

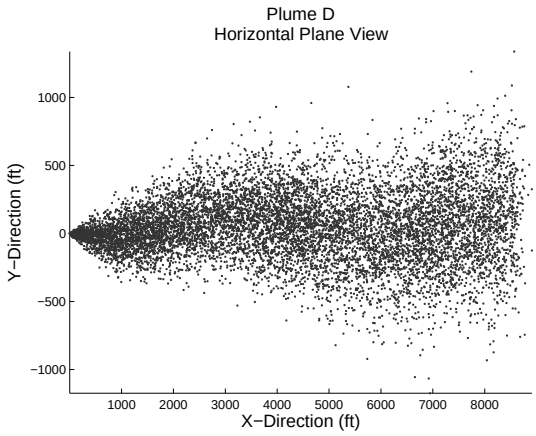


(c) Vertical plane

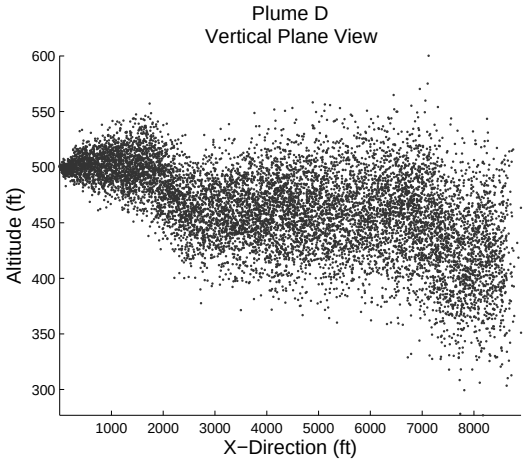
Figure 4.19: 3-D, plume C simulation results. (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

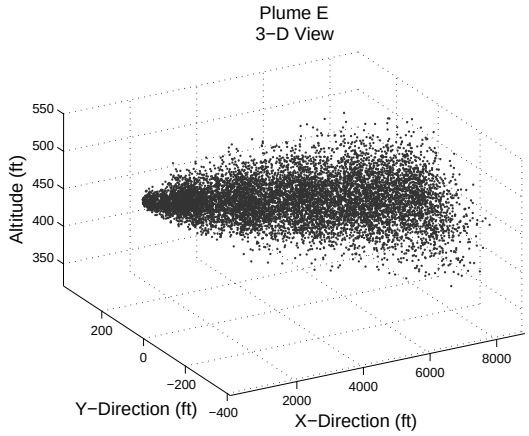


(b) Horizontal plane

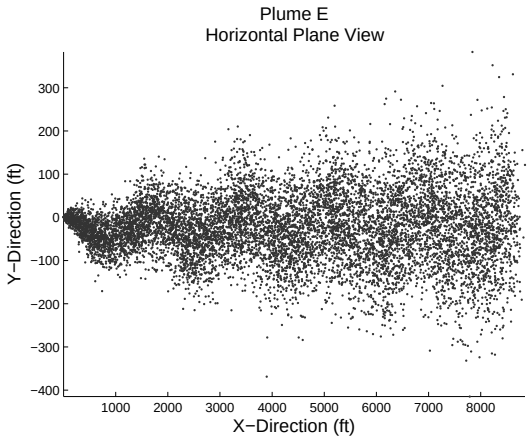


(c) Vertical plane

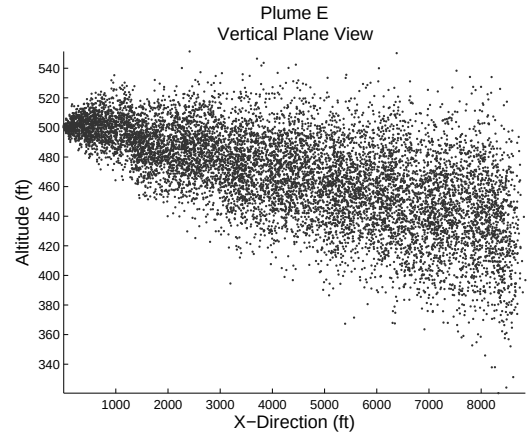
Figure 4.20: 3-D, plume D simulation results. (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.21: 3-D, plume E simulation results. (a) 3-D view (b) Horizontal plane (c) Vertical plane

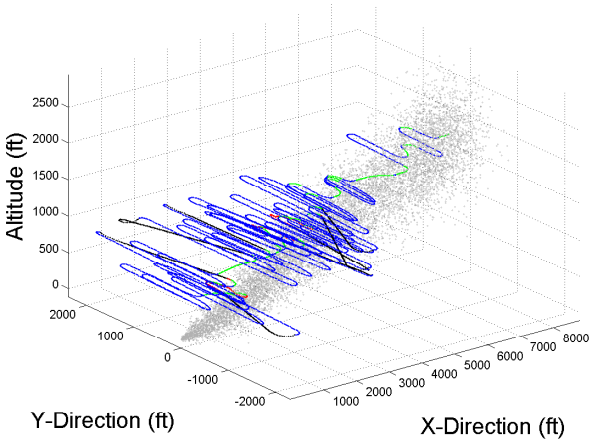
ζ and ΔT_{ct} used for each scenario. Each scenario was tested using a sensor radius of 25 ft. This value was chosen because it made enough detections during preliminary testing to allow the UAV to navigate a simple plume.

Table 4.5: Scenarios tested in initial 3-D study.

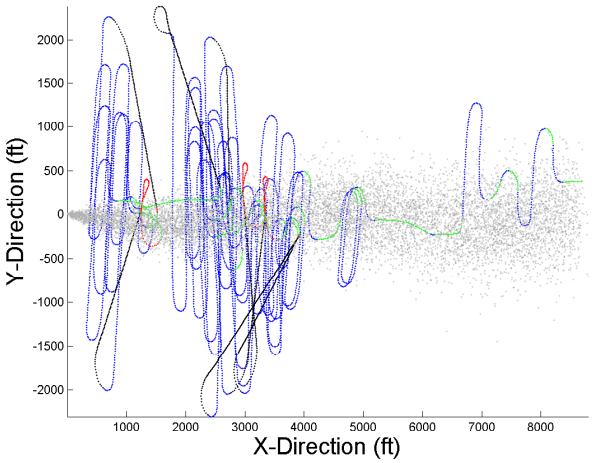
Scenario	ζ (s)	ΔT_{ct} (s)
1	240	20
2	300	20
3	180	20
4	420	20
5	240	10
6	240	30
7	180	10

4.4.1.1 Plume B Results. The testing against plume B revealed ζ values of 300 s and 420 s were unsuccessful, as seen in the results shown in Table 4.6. These values enabled the UAV to stay in the Horizontal Search routine too long (given the plume structure), searching where there was no plume. This was also the case for scenario 6, which used a ζ of 240 s but a ΔT_{ct} of 30 s. The larger ΔT_{ct} allowed for a wider Horizontal Search pattern, again driving the UAV too far outside the plume and decreasing its up wind movement. Such a waste of time is revealed by the decrease of successes and increase in time out failures for the associated scenarios. Figure 4.22 illustrates one of these time out failures (scenario 4, run 3), proving the UAV flew, on many occasions, too far from the plume. This misuse of time resulted in the UAV traveling only to within 640 ft from the source after a flight time of 7,000 s.

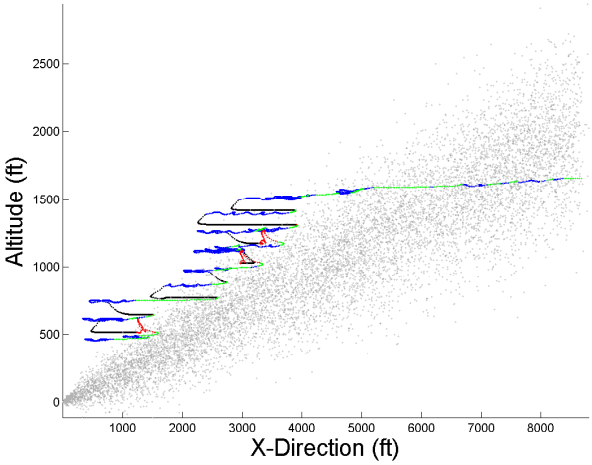
A better way to present the data on the scenarios is graphically. Figure 4.23 illustrates the number of successes per scenario along with the statistics associated with the time it takes the UAV to successfully reach the source. Scenario 7 had the shortest average time, yet the largest standard deviation. Such a result is likely due to it having the highest success rate. Figure 4.24 depicts one of the successful runs (scenario 7, run 6). The UAV flies out of the plume, but not as far as the less successful scenarios mentioned earlier, allowing it to return to the plume quicker. This allowed



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.22: 3-D simulation: unsuccessful navigation of plume B (scenario 4, run 3). (a) 3-D view (b) Horizontal plane (c) Vertical plane

Table 4.6: 3-D initial simulation results when tested against plume B (10 runs).

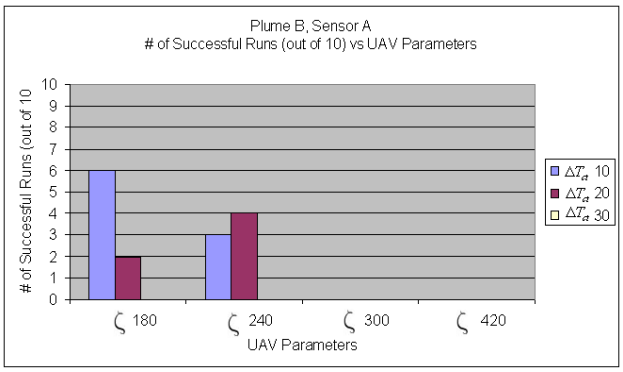
Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
1	4	6	0	6351	481
2	0	9	1	n/a	n/a
3	2	5	3	6660	354
4	0	7	3	n/a	n/a
5	3	5	2	6274	326
6	0	7	3	n/a	n/a
7	6	2	2	5675	652

for a greater portion of time spent in the Tracking routine. Figure 4.25 illustrates a magnified version of the successful termination of the scenario illustrated in Figure 4.24. The UAV gradually decreases its altitude as it flies out of the plume using the Backtrack routine to bring it back to an altitude lower than the last detection.

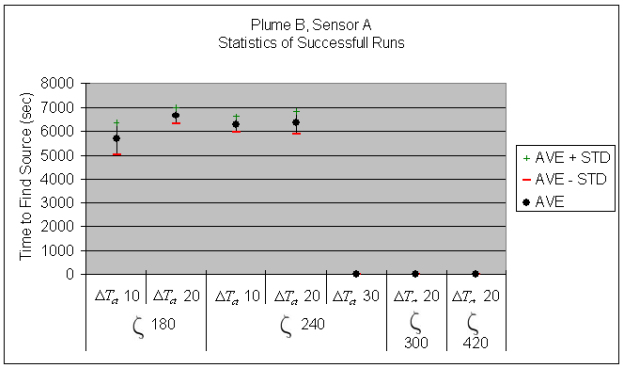
For completeness, Figure 4.26 (scenario 4, run 5) was included to illustrate a failure of finishing the Vertical Search routine without a detection (i.e., losing the plume). This is the lone figure showing this type of failure, as it does not provide much insight and these type of failures have the same appearance.

4.4.1.2 Plume C Results. The testing against plume C revealed similar results among the scenarios as the testing against plume B. This can be seen by the data from plume C presented in Table 4.7. Once again, scenarios with values of 300 s and 420 s for ζ or a ΔT_{ct} value of 30 s were poor performers. Figure 4.27 illustrates a time out failure (scenario 2, run 8) where the UAV only gets to within 3,836 ft from the source, due to the same problems discussed in Section 4.4.1.1.

The graphical representation of the successes and associated statistics that is given in Figure 4.28, is almost identical to the results of plume B testing. The quickest time to source came from scenario 7, run 7 with a time of 4,468 s. This run is shown in Figure 4.29. As with plume B results, the plume C results are positive when simply



(a) Number of successful runs out of 10

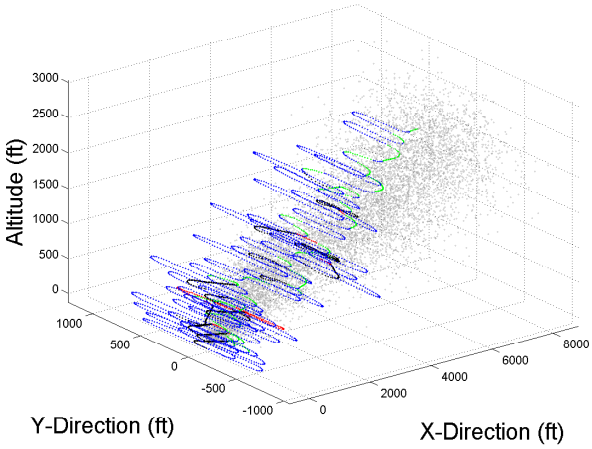


(b) Statistics of successful runs

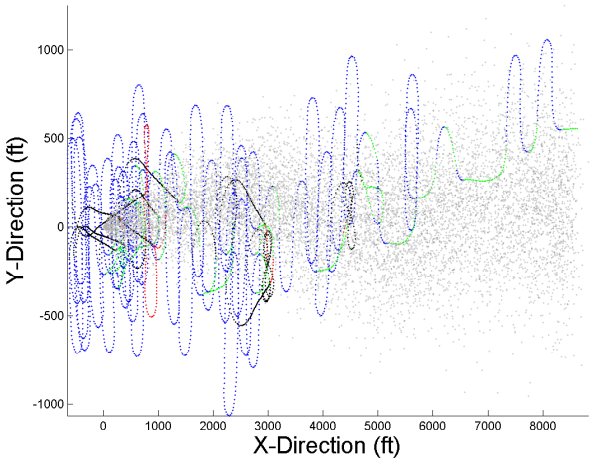
Figure 4.23: 3-D, plume B simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

Table 4.7: 3-D initial simulation results when tested against plume C (10 runs).

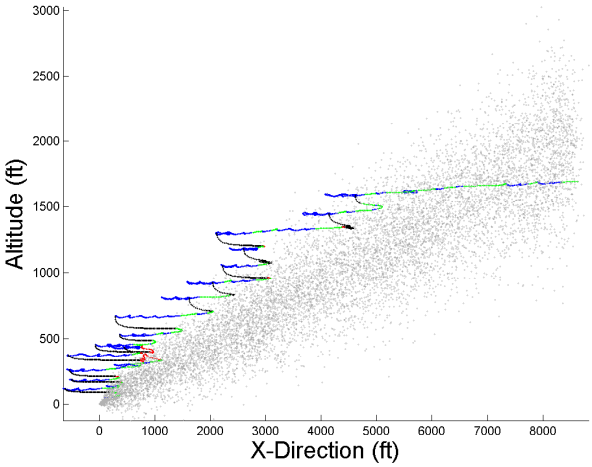
Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
1	5	4	1	6185	488
2	0	9	1	n/a	n/a
3	5	5	0	6183	299
4	0	8	2	n/a	n/a
5	8	2	0	5328	657
6	0	9	1	n/a	n/a
7	9	0	1	4651	523



(a) 3-D view

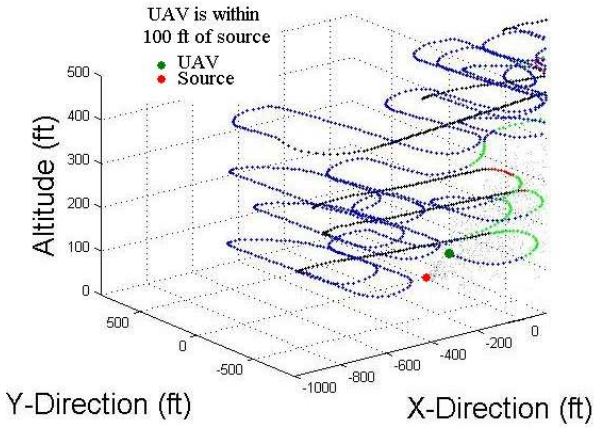


(b) Horizontal plane

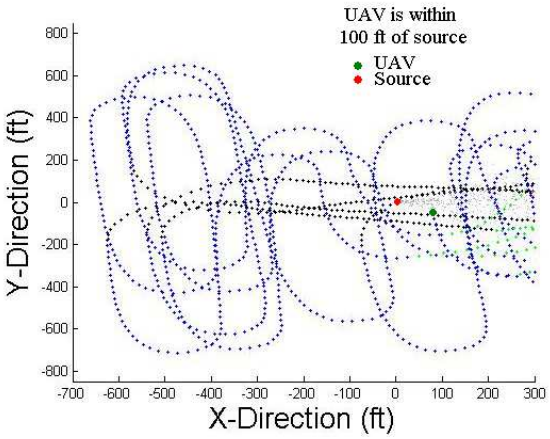


(c) Vertical plane

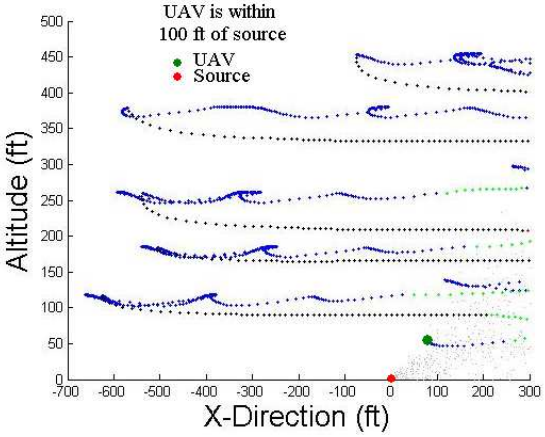
Figure 4.24: 3-D simulation: successful navigation of plume B (scenario 7, run 6). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

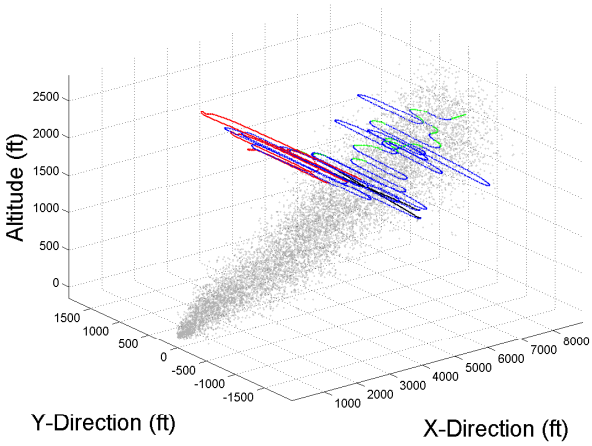


(b) Horizontal plane

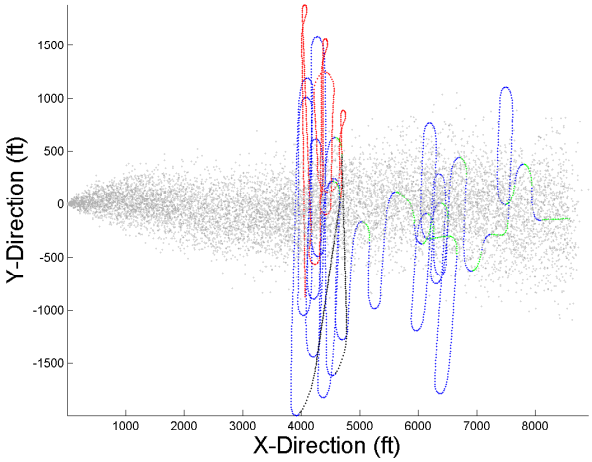


(c) Vertical plane

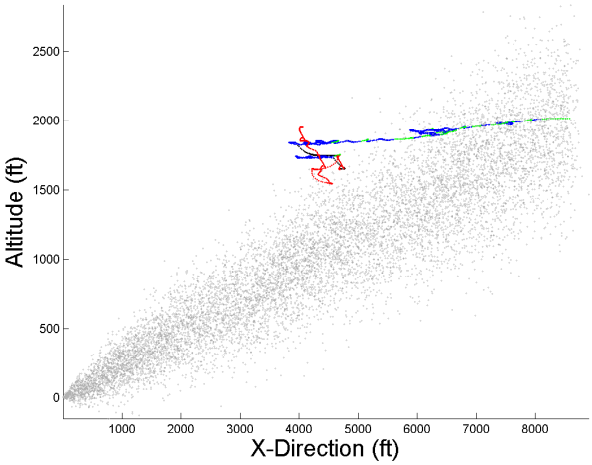
Figure 4.25: 3-D simulation: successful navigation of plume B (scenario 7, run 6). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

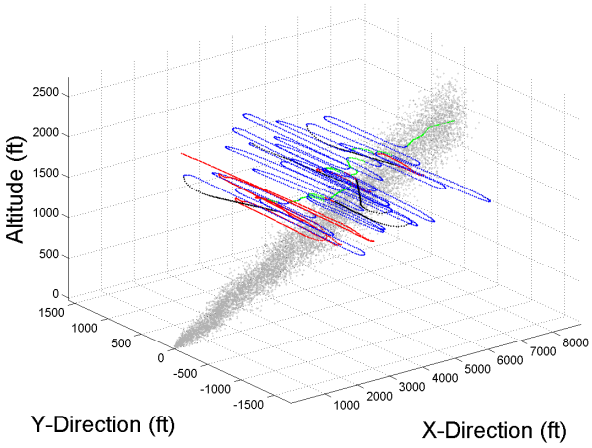


(b) Horizontal plane

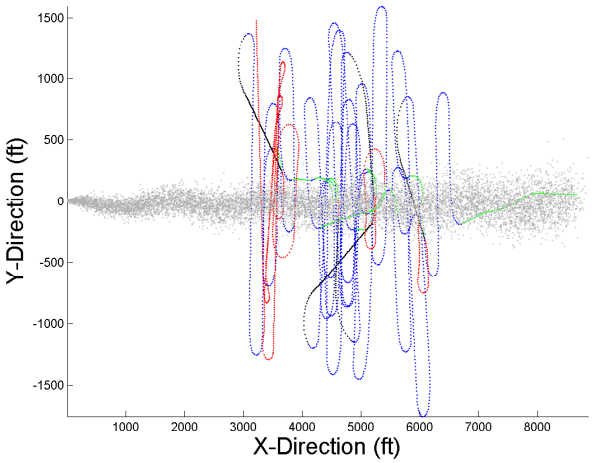


(c) Vertical plane

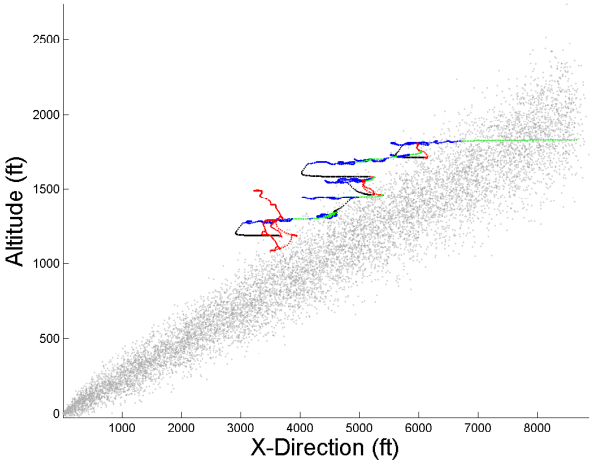
Figure 4.26: 3-D simulation: unsuccessful navigation (losing the plume) of plume B (scenario 4, run 5). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

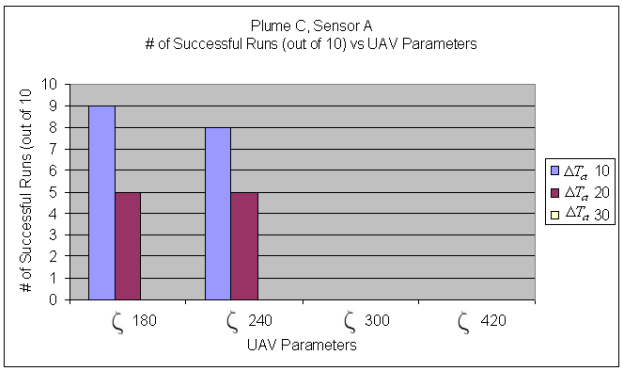


(b) Horizontal plane

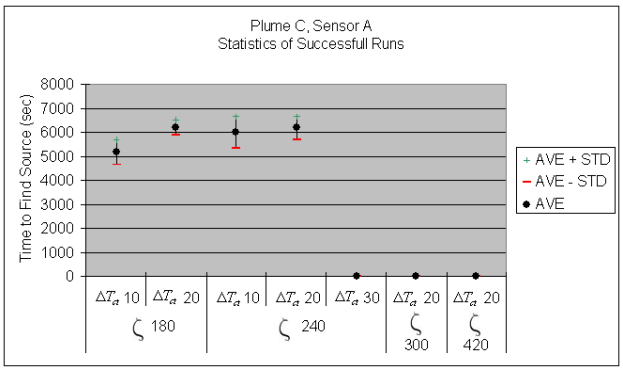


(c) Vertical plane

Figure 4.27: 3-D simulation: unsuccessful navigation of plume C (scenario 2, run 8). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) Number of successful runs out of 10

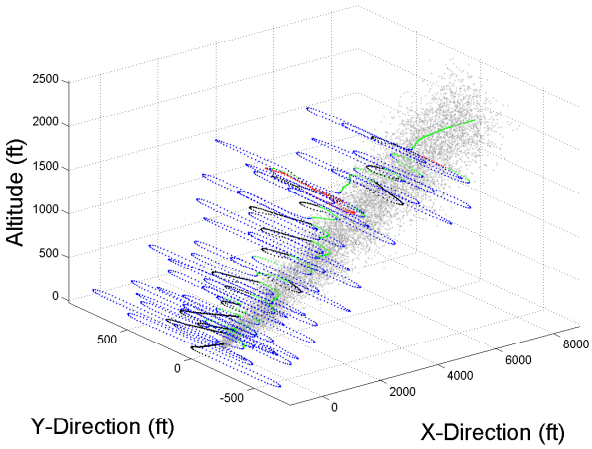


(b) Statistics of successful runs

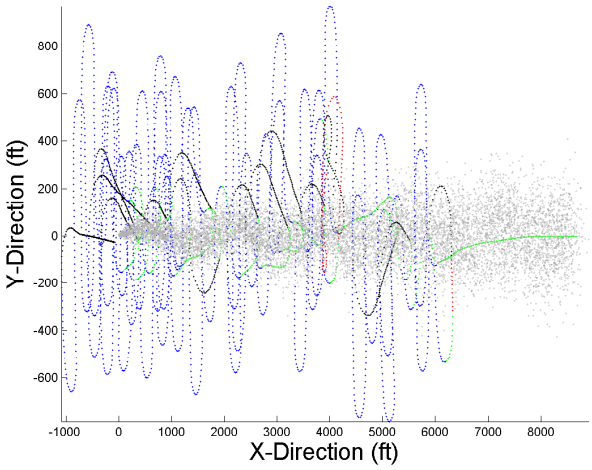
Figure 4.28: 3-D, plume C simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

studying the success rates. However, the time it takes for the UAV to reach the source (9,000 ft away) is not optimal. Taking approximately 1.5 hours to traverse 9,000 ft in an aircraft moving at a rate of 12 to 30 $\frac{\text{ft}}{\text{s}}$ would be unacceptable in an operational system. Such time-to-source statistics can be attributed to the excessive amount of time spent searching outside the plume. Certainly, this is one area of the navigation algorithm that could be improved, at least for these specific plume cases.

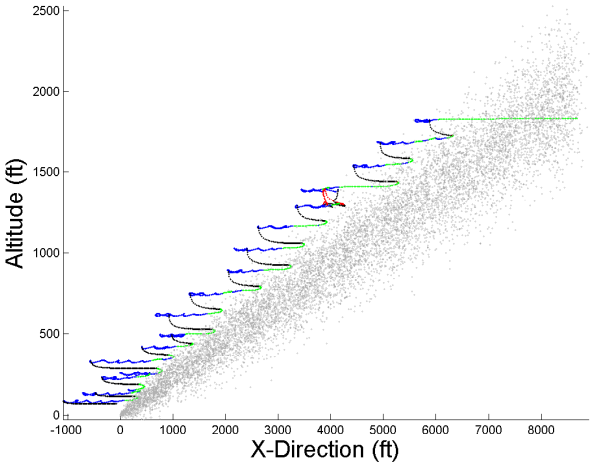
4.4.1.3 Plume D Results. Given the less dynamic nature in the vertical plane of plume D, it was no surprise when the results showed quicker times to reach the source. Table 4.8 gives the results from this testing, revealing all scenarios were successful, even those that included ζ values of 300 s or 400 s or a ΔT_{ct} value of 30 s. To clarify the results, Figure 4.30 illustrates the number of successes and corresponding



(a) 3-D view

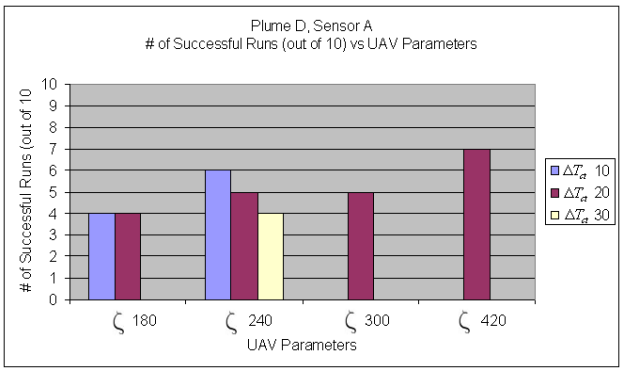


(b) Horizontal plane

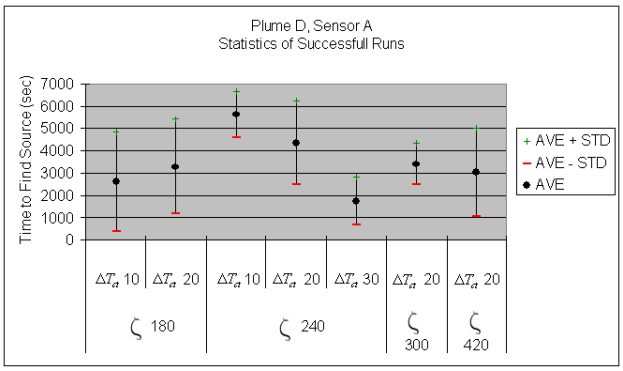


(c) Vertical plane

Figure 4.29: 3-D simulation: successful navigation of plume C (scenario 7, run 7). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) Number of successful runs out of 10



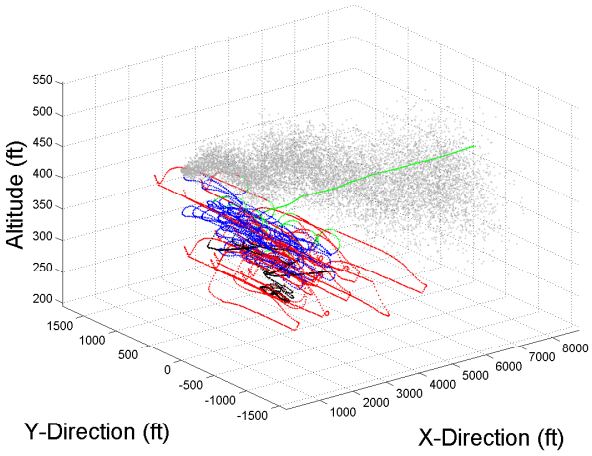
(b) Statistics of successful runs

Figure 4.30: 3-D, plume D simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

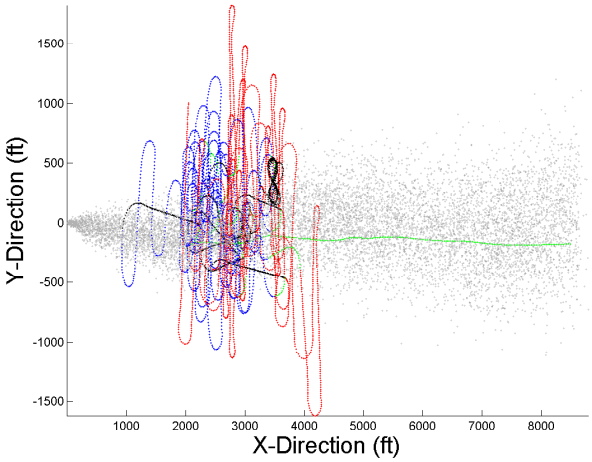
statistics. Oddly enough, the most successful scenario was one that included a ζ of 420 s and the one that was quickest to the source had a ΔT_{ct} value of 30 s. The quickest success (time of 424 s) is shown in Figure 4.31 (scenario 7, run 4). Here, the UAV never has to backtrack, but simply switch between the Tracking and Horizontal Search routines.

As the navigation algorithm spends a majority of its time searching in the horizontal plane, it is better suited for more vertically stable plumes. Therefore, having increased ζ values or ΔT_{ct} values, potentially, aid in navigating the plume. This is true as long as the UAV is at the same altitude as the bulk of the plume.

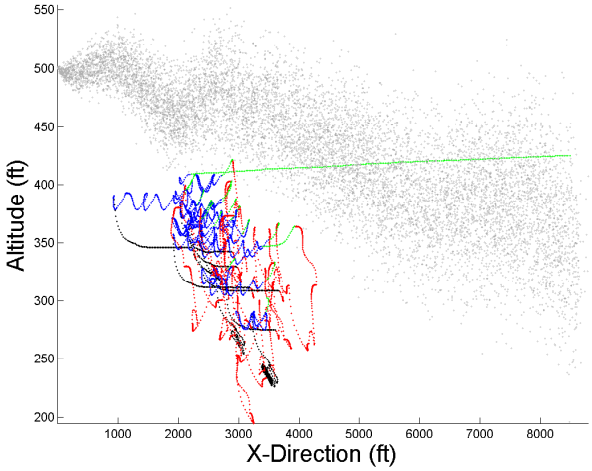
The failures are again due to spending too much time outside of the plume. However, instead of flying over the plume, as is the case with plumes B and C, the



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.31: 3-D simulation: successful navigation of plume D (scenario 7, run 4). (a) 3-D view (b) Horizontal plane (c) Vertical plane

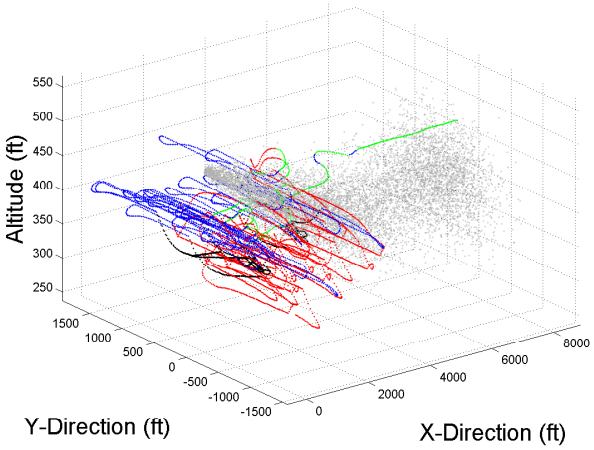
Table 4.8: 3-D initial simulation results when tested against plume D (10 runs).

Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
1	5	5	0	4360	1883
2	5	5	0	3411	938
3	4	6	0	3287	2146
4	7	3	0	3024	1969
5	6	4	0	5622	1031
6	4	6	0	1728	1074
7	4	6	0	2606	2231

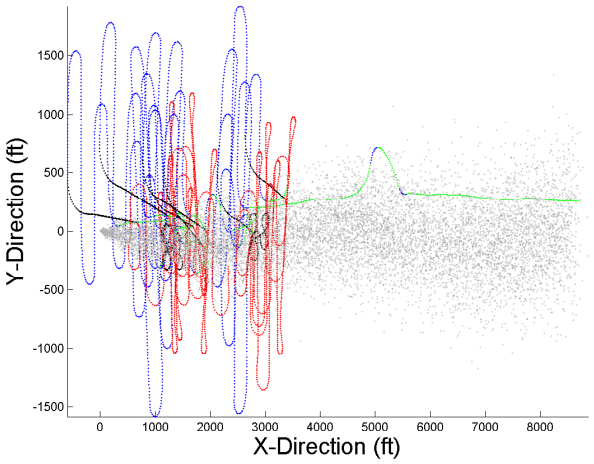
UAV flies under the plume. This can be seen in Figure 4.32 (scenario 1, run 5), illustrating a time-out failure. When looking at the UAV's trajectory in Figure 4.32, it appears the UAV is rapidly changing altitudes in comparison to the trajectories seen in the plume B and C cases. However, the scale of the altitude changed are much lower for the plume D and E cases. These fluctuations are due to losing altitude during a turn and regaining it when the turn is complete, as discussed in Chapter III.

4.4.1.4 Plume E Results. As was the case with plume D, plume E has less vertical dynamics than plumes B and C. The main difference between plumes D and E is plume E has a quicker meander in the horizontal plane, potentially making it harder to navigate. In fact, this appears to be the cause for the slightly worse results for plume E than plume D, as can be seen in Table 4.9. The significant change was the drop in success rate of scenario 4 from plume D to E testing. Figure 4.33 illustrates one of these time out failures (scenario 4, run 8), as the UAV travels to 4,704 ft from the source.

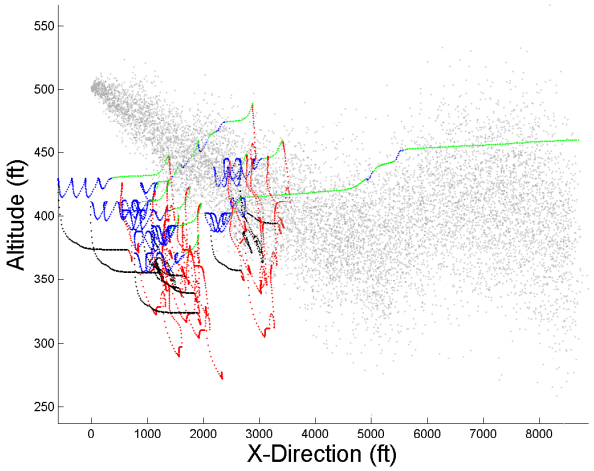
Figure 4.34 better illustrates the results from Table 4.9, making the larger size of the standard deviations from plume D to plume E more evident. Although these results were slightly worse than plume D (except for scenario 7), this testing did



(a) 3-D view

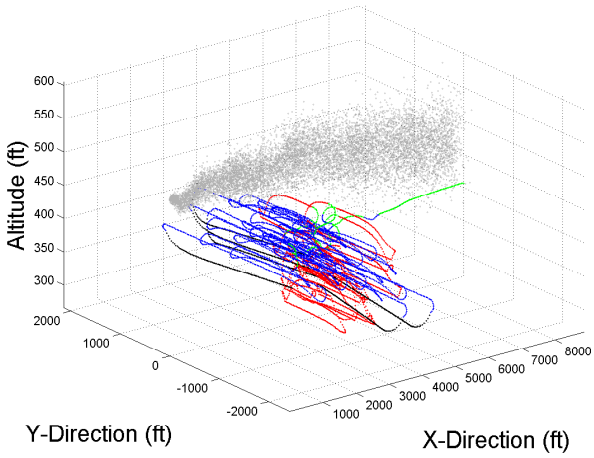


(b) Horizontal plane

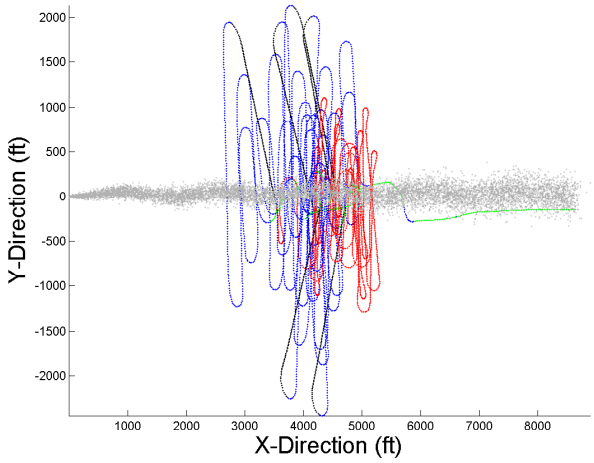


(c) Vertical plane

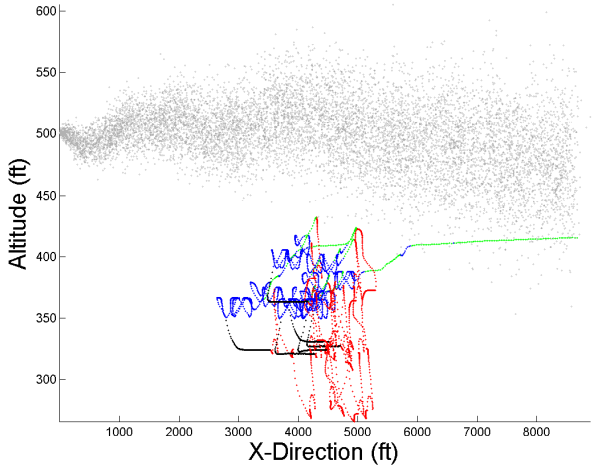
Figure 4.32: 3-D simulation: unsuccessful navigation of plume D (scenario 1, run 5). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.33: 3-D simulation: unsuccessful navigation of plume E (scenario 1, run 5). (a) 3-D view (b) Horizontal plane (c) Vertical plane

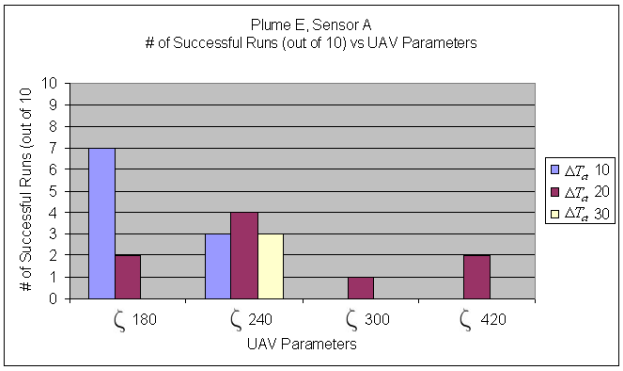
Table 4.9: 3-D initial simulation results when tested against plume E (10 runs).

Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
1	4	6	0	4841	2413
2	1	8	1	5447	n/a
3	2	8	0	3352	2191
4	2	8	0	4978	1759
5	3	7	0	3659	3192
6	3	7	0	5261	1199
7	7	3	0	3957	1952

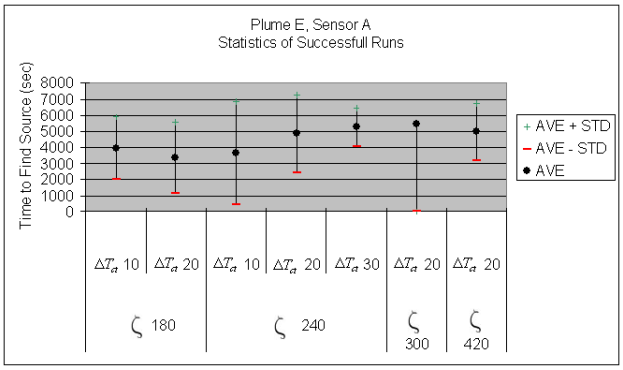
produce the quickest time-to-source of all the runs in the initial simulation study. This run (scenario 7, run 4) reached the source in 398 s, and is illustrated in Figure 4.35.

4.4.1.5 Discussion of Results. From the results of the initial study, an additional set of scenarios was developed to increase the UAV's success rate and decrease the time to locate the source. Scenarios 1, 3, 5, and 7 were the most robust algorithms in the study. They performed much better than the other scenarios when navigating plumes B and C, and performed just as well when navigating plumes D and E. These four scenarios incorporated the smallest values given to ζ (180 s or 240 s) and ΔT_{ct} (10 s or 20 s) in the study. These smaller values decrease both the width and length of the search pattern generated from the Horizontal Search routine, therefore, decreasing the time out of the plume. The best performing scenario (7) had a ζ of 180 s and a ΔT_{ct} of 10 s. Table 4.10 summarizes this scenario's results for the initial simulation study. It is important to note that all scenarios under this testing incorporated one sensor size (25 ft). As a result, ζ values of 180 and 240 s and ΔT_{ct} values of 10 and 20 s were used as the foundation for the scenarios of the final study.

4.4.2 Final Study. In order to further test the robustness of the initial study's successful scenarios, the final study incorporated both a 25 ft and 40 ft sensor (altering the apparent density of the plume). The other component of the algorithm



(a) Number of successful runs out of 10

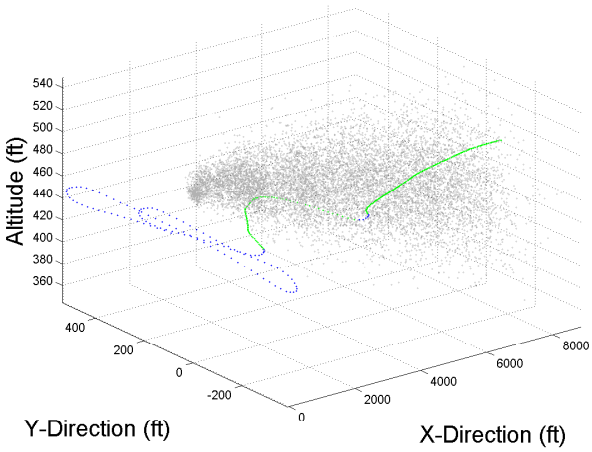


(b) Plume E, Statistics of successful runs

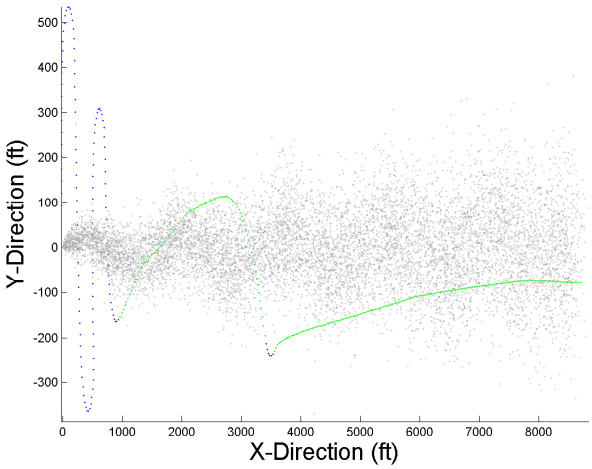
Figure 4.34: 3-D, plume E simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

Table 4.10: Summary of the most successful navigation algorithm for initial study ($\zeta = 180$ s, $\Delta T_{ct} = 10$, sensor size = 25 ft).

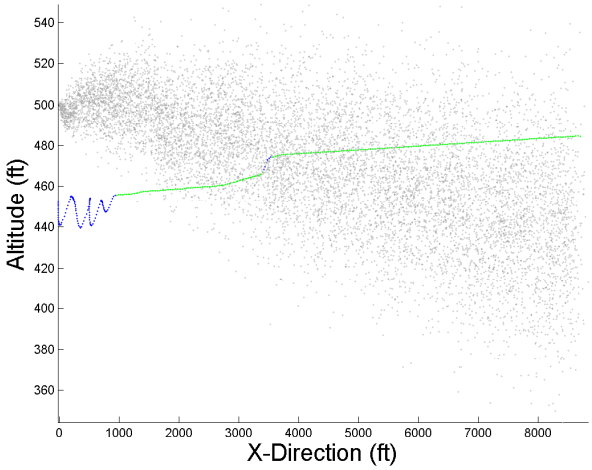
Plume	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
B	6	2	2	5675	652
C	9	0	1	4651	523
D	4	6	0	2606	2231
E	7	3	0	3957	1952



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.35: 3-D simulation: successful navigation of plume E (scenario 7, run 4). (a) 3-D view (b) Horizontal plane (c) Vertical plane

varied in this study was the Backtrack routine. The new Backtrack routine commands the UAV to go back to the exact location of the last detection and not a set altitude below that location. In the cases with plumes D and E, the UAV exits below the plume. The original Backtrack routine, therefore, places the UAV under the plume, causing it to conduct a greater percentage of the Vertical Search routine before detecting the plume. Such additional searching lengthens the UAV's total flight time, causing the simulations for plumes D and E to fail by running out of time. The list of scenarios for the final study are given in Table 4.11 (orig = original Backtrack routine, new = new Backtrack routine).

Table 4.11: Scenarios tested in final 3-D study.

Scenario	ζ (s)	ΔT_{ct} (s)	Backtrack Routine	Sensor Size (ft)
8	180	20	orig	40
9	180	10	orig	40
10	240	20	orig	40
11	240	10	orig	40
12	180	20	new	25
13	180	10	new	25
14	240	20	new	25
15	240	10	new	25
16	180	20	new	40
17	180	10	new	40
18	240	20	new	40
19	140	10	new	40

4.4.2.1 Plume B and C Results. The final testing results against plumes B and C were once again very similar, with the scenarios again performing slightly better against plume C than plume B. The results for plumes B and C are given in Table 4.12 and 4.13, respectively. It is worth noting that the UAV never located the source within the 7,000 s time frame while using the new Backtrack routine. This is not too surprising as the UAV (with the original Backtrack routine) took 5,000 s to 6,500 s to find the source. Applying the new Backtrack routine against a rising plume forces the UAV to perform more of the Vertical Search routine before detecting the plume. Figure 4.36 illustrates one of the unsuccessful trajectories for

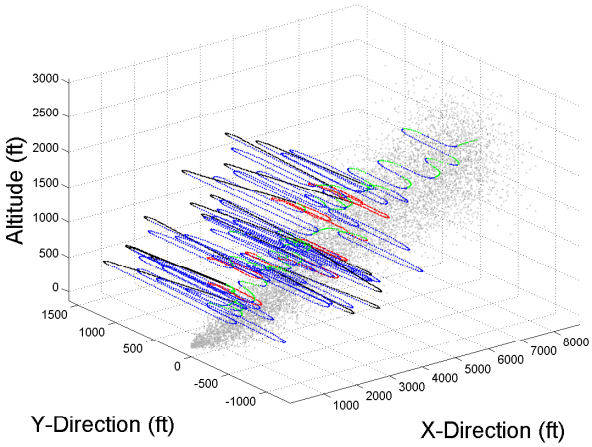
the plume B testing (scenario 12, run 2), where the UAV comes within 1,172 ft from the source. The other failures for the plume B or C testing have similar characteristics and are not shown.

Table 4.12: 3-D final simulation results when tested against plume B (10 runs).

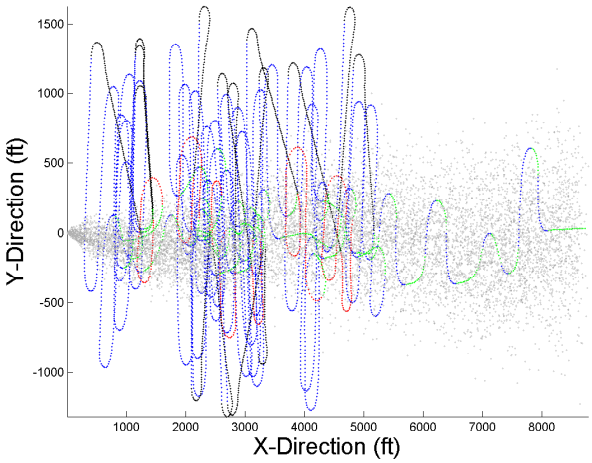
Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
8	3	5	2	6460	382
9	7	1	2	5473	730
10	6	3	1	6446	529
11	5	4	1	5983	501
12	0	9	1	n/a	n/a
13	0	8	2	n/a	n/a
14	0	7	3	n/a	n/a
15	0	6	4	n/a	n/a
16	0	9	1	n/a	n/a
17	0	9	1	n/a	n/a
18	0	9	1	n/a	n/a
19	0	7	3	n/a	n/a

The four successful scenarios for both plumes incorporate the original Backtrack routine and the new 40 ft sensor. These results are slightly better than when tested with the 25 ft sensor (Tables 4.6 and 4.7) and are expected due to the larger sensor having a greater chance of detection. Figures 4.37 and 4.38 give a clearer depiction of the successes and their associated statistics. An example of a successful trajectory against plume B is illustrated by Figure 4.39 (scenario 9, run 3), the UAV locates the source in 4,644 s. The other successes for plumes B and C have similar characteristics and are not shown.

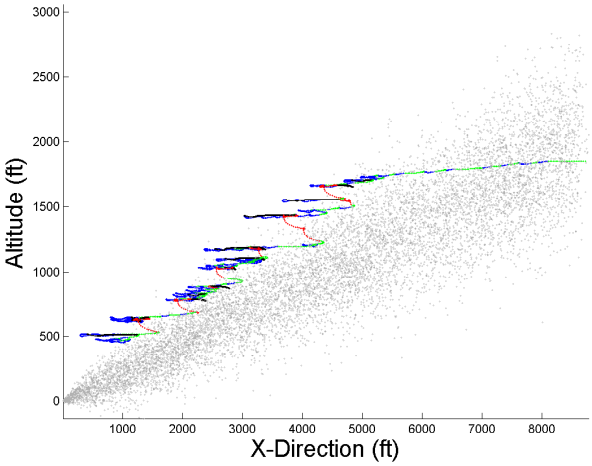
4.4.2.2 Plume D and E Results. The results from the final testing for plumes D and E revealed, once again, that the navigation algorithm was slightly more successful in traversing plume D than plume E. This can be seen by looking at the plume D and E results given in Tables 4.14 and 4.15, respectively. As was the case



(a) 3-D view

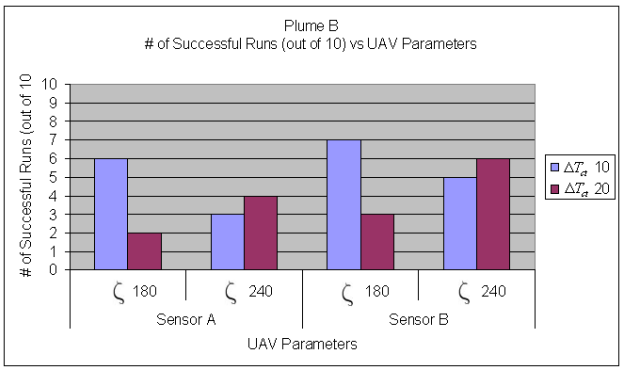


(b) Horizontal plane

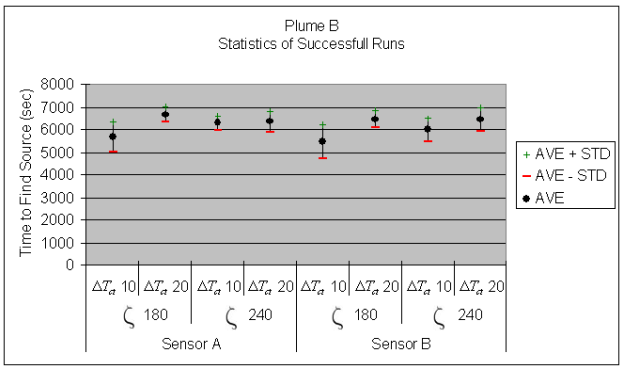


(c) Vertical plane

Figure 4.36: 3-D final simulation study: unsuccessful navigation of plume B (scenario 12, run 2). (a) 3-D view (b) Horizontal plane (c) Vertical plane

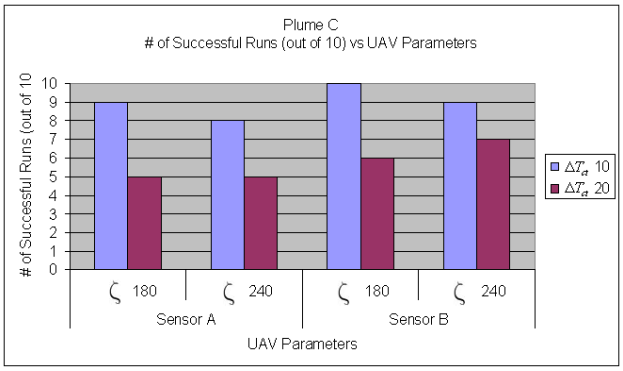


(a) Number of successful runs out of 10

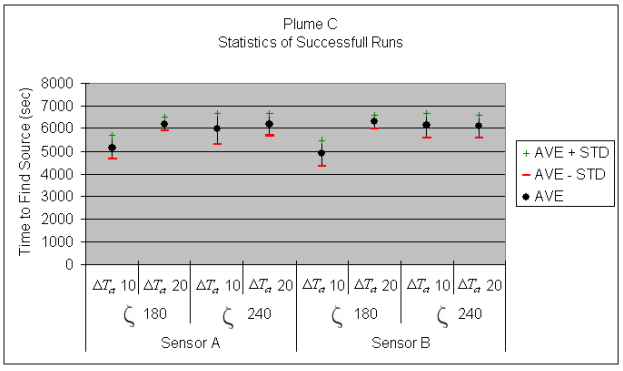


(b) Statistics of successful runs

Figure 4.37: 3-D, plume B final simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

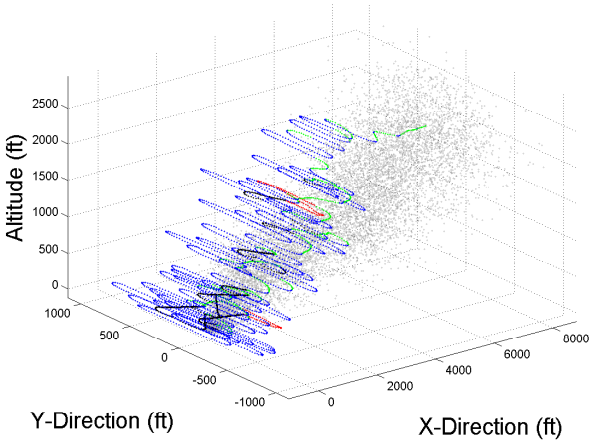


(a) Number of successful runs out of 10

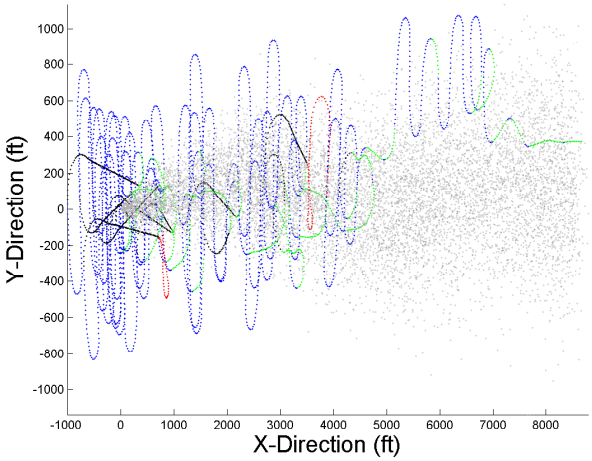


(b) Statistics of successful runs

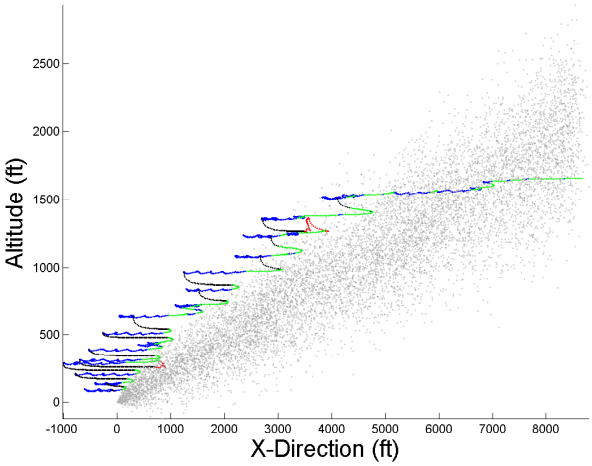
Figure 4.38: 3-D, plume C final simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.39: 3-D final simulation study: successful navigation of plume B (scenario 9, run 3). (a) 3-D view (b) Horizontal plane (c) Vertical plane

Table 4.13: 3-D final simulation results when tested against plume C (10 runs).

Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
8	6	3	0	6276	306
9	10	0	0	4921	570
10	7	2	1	6086	473
11	9	1	0	6130	514
12	0	8	2	n/a	n/a
13	0	10	0	n/a	n/a
14	0	8	2	n/a	n/a
15	0	7	3	n/a	n/a
16	0	9	1	n/a	n/a
17	0	10	0	n/a	n/a
18	0	9	1	n/a	n/a
19	0	10	0	n/a	n/a

with the final study of plumes B and C, the use of a larger sensor (i.e., higher density plume) resulted in a modest improvement of both the successes and their statistics. Figures 4.40 and 4.41 graphically illustrate this information. Figure 4.42 depicts a successful run in the plume E testing (scenario 9, run 4), with the UAV locating the source of the plume in 320 s. An example of a time out failure (Scenario 9, run 6) is shown in Figure 4.43, where the UAV travelled to within 1028 ft of plume E's source.

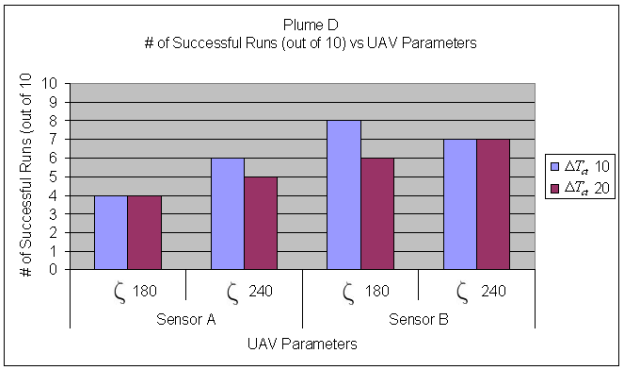
The use of the new Backtrack routine did not hinder the tracking of these plumes as it did with plumes B and C. However, it served no benefit as the results were similar to those when the original Backtrack routine was used, as given is Tables 4.8 and 4.9. Figures 4.44 and 4.45 graphically illustrate this information. Figures 4.46 and 4.47 illustrate successful and unsuccessful tests against plume D. The successful run (scenario 17, run 3) had with a time-to-source of 484 s and the unsuccessful run (scenario 17, run 7) had a time-out failure ending the simulation at a distance to source of 2329 ft from the source.

Table 4.14: 3-D final simulation results when tested against plume D (10 runs).

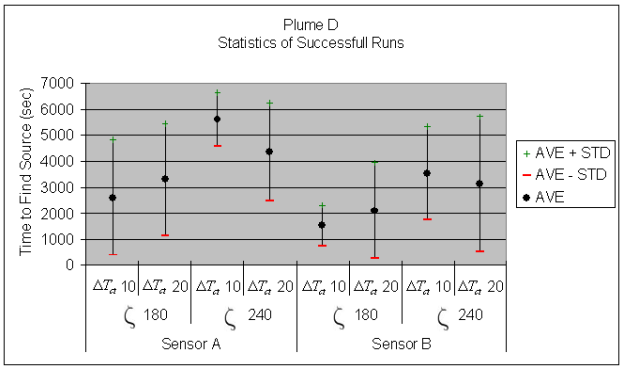
Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
8	6	4	0	2089	1844
9	10	0	0	1532	769
10	7	2	1	3131	2588
11	7	3	0	3534	1787
12	5	5	0	3230	2042
13	8	2	0	3508	2356
14	6	4	0	3060	1182
15	4	5	1	2885	2585
16	5	5	0	3252	2213
17	8	2	0	2334	1416
18	6	4	0	2305	1916
19	4	6	0	1737	1582

Table 4.15: 3-D final simulation results when tested against plume D (10 runs).

Scenario	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
8	6	4	0	2950	1383
9	9	1	0	3947	2084
10	3	7	0	3673	2944
11	6	4	0	3751	2281
12	2	8	0	4674	1773
13	5	5	0	4047	2274
14	3	6	1	5624	678
15	6	4	0	5013	1178
16	5	5	0	3852	2572
17	5	5	0	2454	1813
18	4	6	0	3204	2566
19	6	4	0	4677	2553

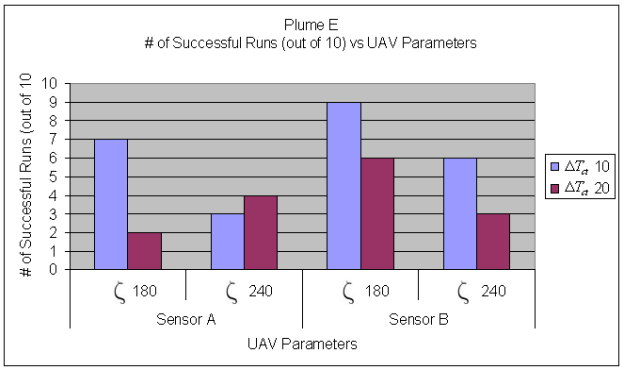


(a) Number of successful runs out of 10

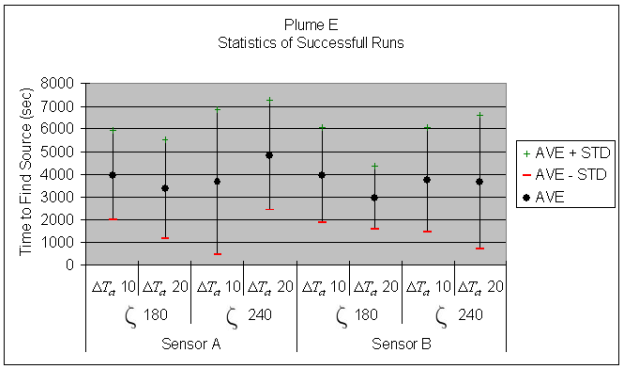


(b) Statistics of successful runs

Figure 4.40: 3-D, plume D final simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.

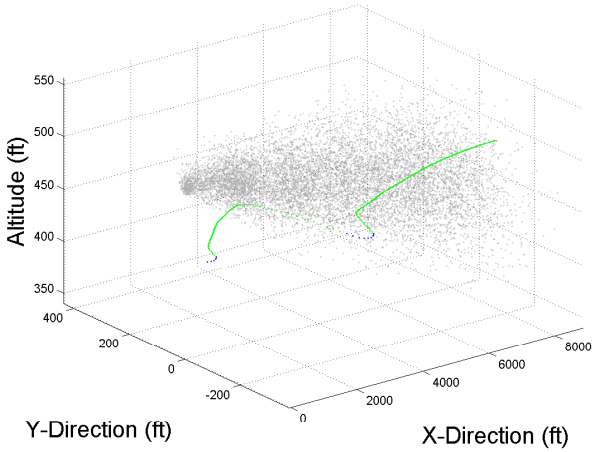


(a) Number of successful runs out of 10

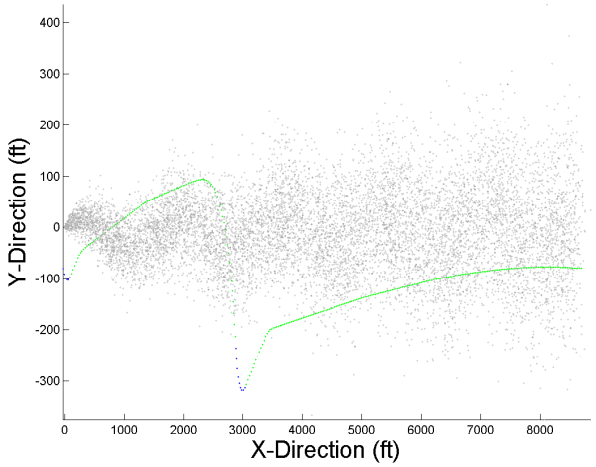


(b) Statistics of successful runs

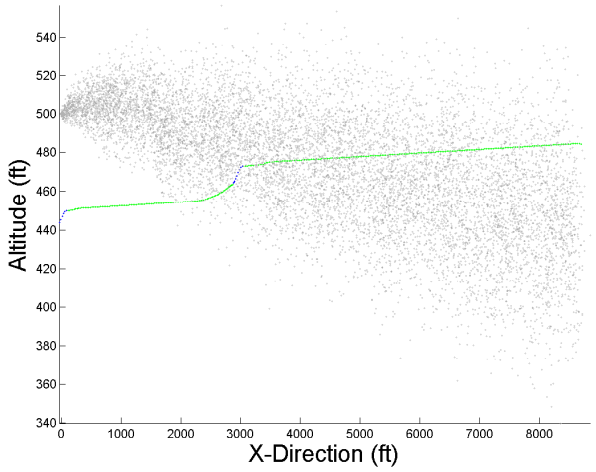
Figure 4.41: 3-D, plume E final simulation results. (a) Number of successful runs out of 10. (b) Statistics of successful runs.



(a) 3-D view

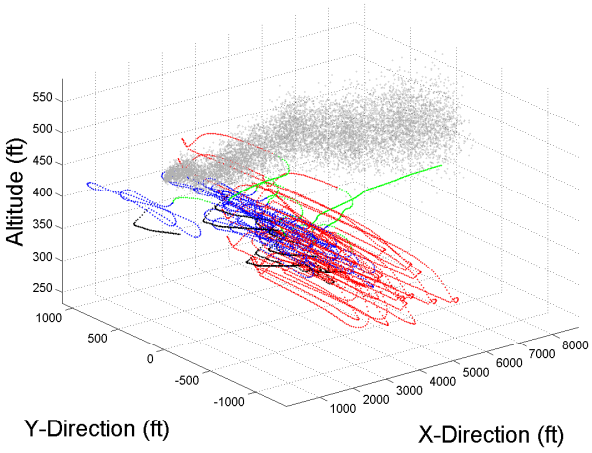


(b) Horizontal plane

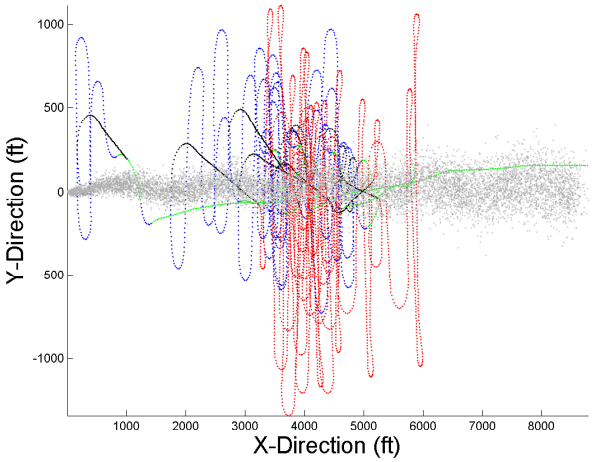


(c) Vertical plane

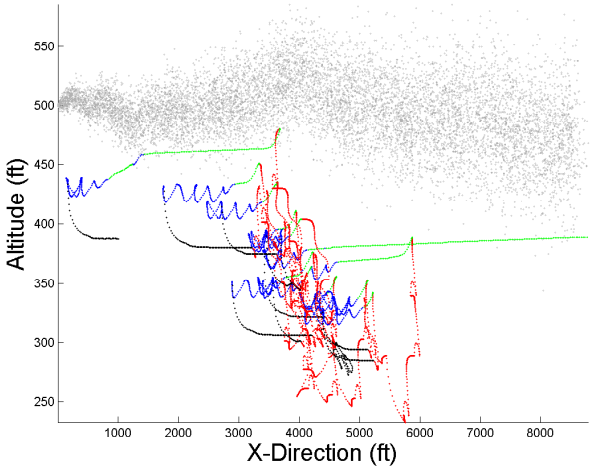
Figure 4.42: 3-D final simulation study: successful navigation of plume E (scenario 9, run 4). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view

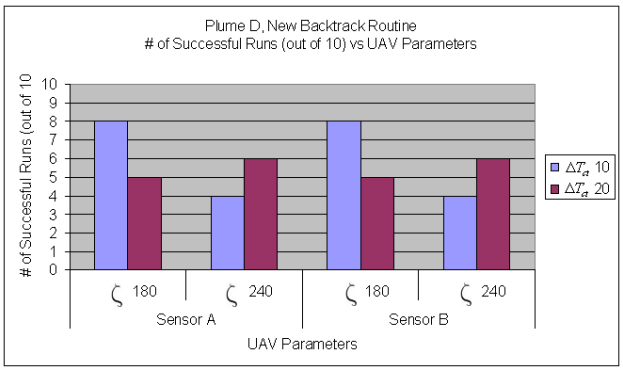


(b) Horizontal plane

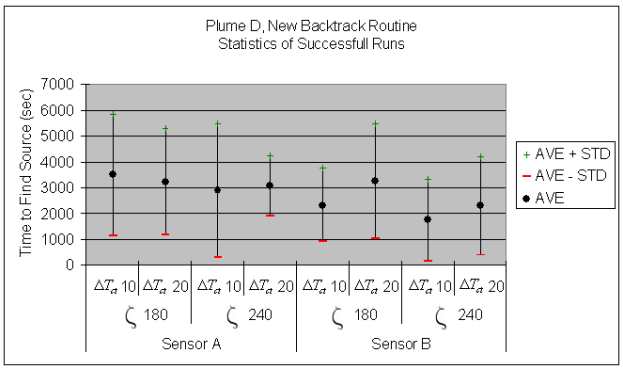


(c) Vertical plane

Figure 4.43: 3-D final simulation study: unsuccessful navigation of plume E (scenario 9, run 6). (a) 3-D view (b) Horizontal plane (c) Vertical plane

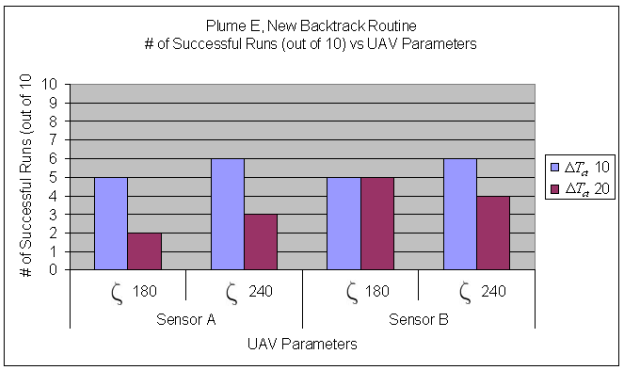


(a) Number of successful runs out of 10

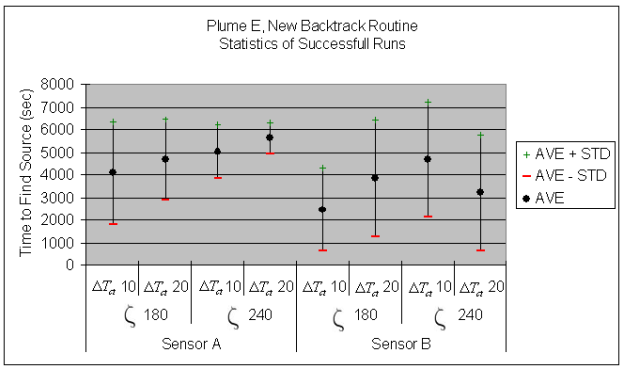


(b) Statistics of successful runs

Figure 4.44: 3-D, plume D final simulation results (new Backtrack routine). (a) Number of successful runs out of 10. (b) Statistics of successful runs.

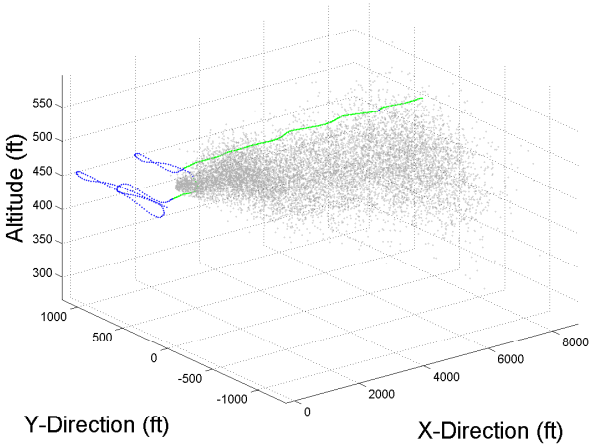


(a) Number of successful runs out of 10

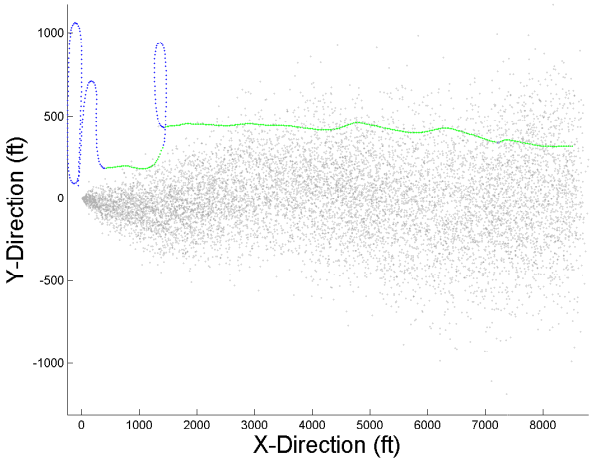


(b) Statistics of successful runs

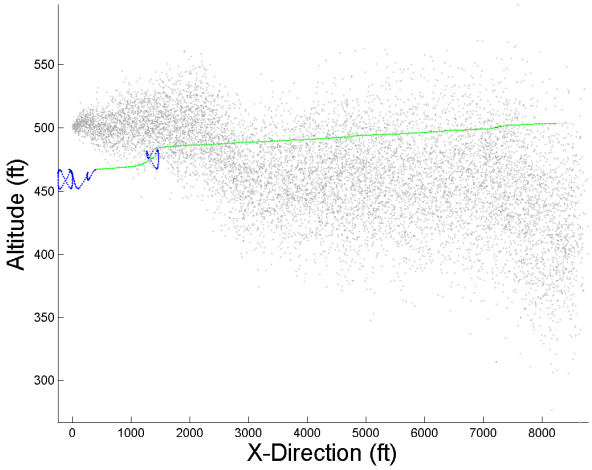
Figure 4.45: 3-D, plume E final simulation results (new Backtrack routine). (a) Number of successful runs out of 10. (b) Statistics of successful runs.



(a) 3-D view

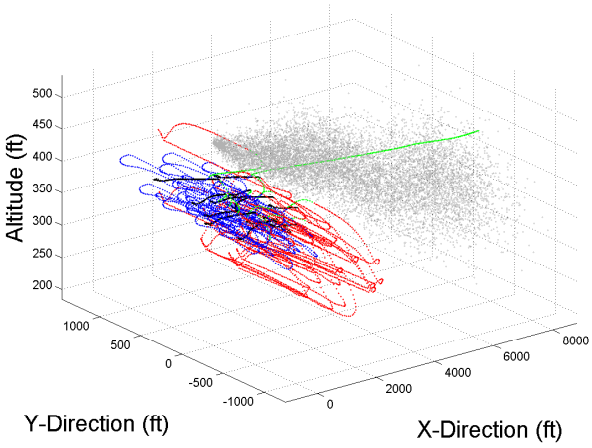


(b) Horizontal plane

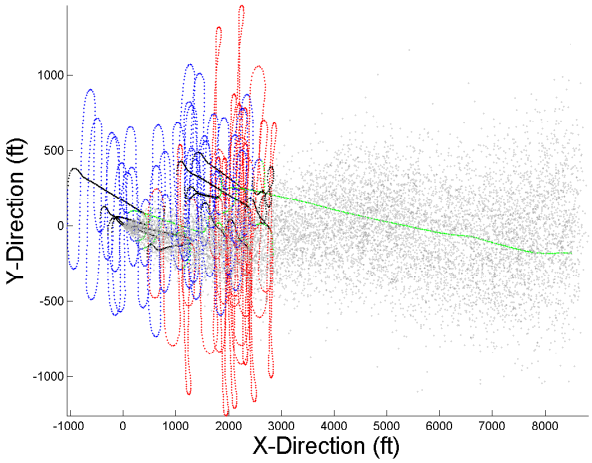


(c) Vertical plane

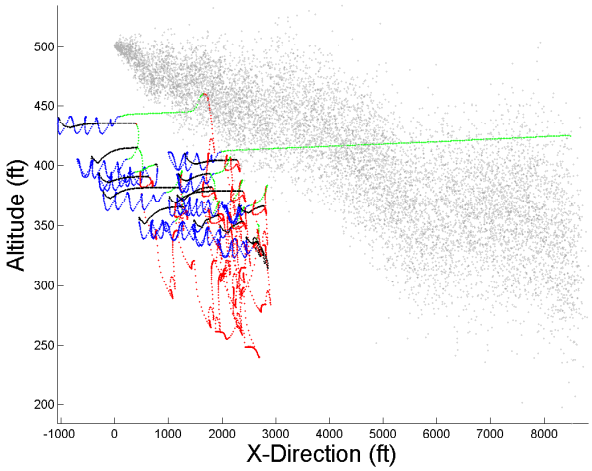
Figure 4.46: 3-D final simulation study (new Backtrack routine): successful navigation of plume D (scenario 17, run 3). (a) 3-D view (b) Horizontal plane (c) Vertical plane



(a) 3-D view



(b) Horizontal plane



(c) Vertical plane

Figure 4.47: 3-D final simulation study (new Backtrack routine): unsuccessful navigation of plume D (scenario 17, run 7). (a) 3-D view (b) Horizontal plane (c) Vertical plane

4.4.2.3 Discussion of Results. The results of this study further proved that decreasing values of ζ and ΔT_{ct} (preferably together) improves both the success rate and time-to-source. The most successful scenarios (9, 13, and 17) had a ζ value of 180 s and a ΔT_{ct} value of 10 s. These values resulted in a decrease in the amount of time the UAV spent outside the plume. The results for the navigation algorithm with these values (with a 25 ft sensor) are shown in Table 4.16. It performs better with the new Backtrack routine against plumes D and E, but does much worse against plumes B and C. The poor performance against plumes B and C led to the conclusion that the original Backtrack routine is more robust. Substandard performance against plumes B and C was due to additional time needed in searching for the plume. This results in an increase in time required to reach the source. This increase in the time-to-source caused a decrease in the success rate.

Table 4.16: Summary of the most successful navigation algorithm implementing new Backtrack routine ($\zeta = 180$ s, $\Delta T_{ct} = 10$, sensor size = 25 ft).

Plume	Successes	Time Out Failures	Lost Plume Failures	Average Time of Successes (s)	Stdev
B	0	8	2	n/a	n/a
C	0	10	0	n/a	n/a
D	8	2	0	3508	2356
E	5	5	0	4047	2274

The new Backtrack routine performed poorly on plumes B and C, as it caused the UAV to spend an increased amount of time in the Vertical Search routine. This caused a majority of the plume B and C simulations to fail due to not locating the source in the allotted timeframe. When this new routine was tested on plumes D and E, the navigation algorithm performed about the same. As a result, the original routine outperforms the new routine and is the obvious preference. Further analysis of the results from this study is presented in Chapter V.

4.5 Summary

The results from the 2-D simulation showed promising results with the use of an STM to aid in tracking a chemical plume. This is evident as the moth wind tunnel experiments exceeded the simulation success rate by only a small margin. This set the foundation for the 3-D UAV navigation algorithm, which worked well depending on the values set for ζ and ΔT_{ct}. The overall performance of the UAV was better when using the original Backtrack routine (designed in Chapter III). These simulation studies merely scratched the surface on presenting the true capabilities of the navigation algorithms developed. From this point there is much more testing and improvement that can be done, some of which are discussed in Chapter V.

V. Conclusions, Contributions, and Recommendations

Given the thorough analysis and discussions of the simulation results presented in Chapter IV, this chapter gives a concise set of conclusions for both the 2-D and 3-D navigation algorithms. In addition, certain aspects of the navigation algorithms and simulation designs developed in this thesis effort do warrant further explanation which is given in Section 5.2. Lastly, this chapter does provide suggested recommendations for continued research in the area of odor-based navigation.

5.1 Conclusions

5.1.1 2-D Navigation. The results of the 2-D simulations and achievements of the navigation algorithm are critiqued based on:

1. Whether the trajectories and success rates were indicative of MSexta wind tunnel tests.
2. Whether or not the navigation algorithm was successful in tracking the plume.

The flight profiles, time-to-source and success rates of the algorithm are most representative of the moth wind tunnel tests (Section 4.3 of Chapter IV) when the sensor size was set to 0.3 cm. The "moth" located the source 64% - 71% of the time and did so with an average time between 9.09 s and 7.31 s, depending on the plume it was tested against. The success of the algorithm, given the variety of plume densities tested (varying sensor sizes) was positive. As the plume density increased (increasing sensor size), the percentage of "moths" locating the source increased, boasting up to an 88% success rate. In addition, the time-to-source decreased as did its associated standard deviation, the best average time being 2.67 s with a standard deviation of 1.78 s. Obviously as the plume gets more dense, the easier it is to detect, reducing the amount of time needed to search for the plume once it is lost. However, one must be careful to recognize that generating a simulated plume is the weak link in the development of computer-based test and evaluation of odor-based navigation schemes. Given the accomplishments of the algorithm, which is heavily dependent on the STM

fuzzy controller (Section 3.2.4 of Chapter III), there is merit to the hypothesis that moths use other means to navigate a pheromone plume besides that of maneuvering based on single detections. The success of this algorithm (success rate of at least 63%) provided a solid foundation from which to build the 3-D algorithm.

5.1.2 3-D Navigation. For the 3-D simulation studies (Section 4.4 of Chapter IV), there was a mixture of success rates in locating the source. Both the successes and failures gave insight into how to more efficiently navigate a plume. As one might expect, limiting the amount of time the UAV spends outside of the plume is extremely important. This was validated by the increased success resulting from decreasing values of ζ and ΔT_{ct}. The smaller these values become, the narrower the horizontal search pattern becomes and the time the UAV stays in this pattern decreases. These factors limit the time spent outside the plume. The best performing scenario (7) had a ζ of 180 s and a ΔT_{ct} of 10 s.

The Backtrack routine worked remarkably well (success rate of at least 40%), even with the drastic variations from plumes B and C to plumes D and E. The new Backtrack routine worked slightly better against plumes D and E than the original routine. None of the navigation algorithms tested with the new Backtrack routine were superior performers. Again, navigation algorithms with values of $\zeta = 180$ s and $\Delta T_{ct} = 10$ performed respectably (success rate of at least 50%). The poor performance against plumes B and C (zero succusses) led to the conclusion that the original Backtrack routine is more robust and was due to additional time needed in searching for the plume. This results in an increase in time required to reach the source.

Another portion of the final study tested the navigation algorithms with a sensor size of 40 ft rather than the previous 25 ft sensor. The results were not surprising, as the trends followed those seen in the 2-D simulations. With the increasing sensor size the percentage of successful runs increased, though not as drastically as in the 2-D cases.

The success (upwards of a 90% success rate) seen in the various navigation algorithms tested in the 3-D simulation studies give promise to the idea of using an STM to navigate a UAV through a chemical plume. As with the 2-D case, the success of the algorithms must be scrutinized. The use of a simulated plume does not guarantee the success of the algorithms when tested against a true chemical plume. It does, however, give support to the techniques used and strengthens the argument for developing a robotic platform to test said techniques.

5.2 Contributions

The underlying theme of the research presented in this thesis, bio-inspired, odor-based navigation, is not totally original. The research used to buttress this topic, as discussed in Chapter II, makes this evident. However, there has been little information found in the open literature that addresses this problem in 3-D. Developing a 3-D simulation environment, incorporating both a realistic dynamics model and a time-varying plume model, is a notable achievement. This environment allows for computer-based testing of plume tracking techniques and gives the Air Force a means to inexpensively test tracking algorithms prior to hardware implementations. This was only one of three developments unique to this thesis investigation that contributes to the research field of odor-based navigation. The other two significant contributions of this work are the STM tracking algorithm and Vertical Search routine, as detailed in Section 3.2.4 of Chapter III.

The STM tracking algorithm used to control horizontal movement of either the simulated moth or UAV is implement the same way in both the 2-D and 3-D simulations. The hypothesis used as the foundation for the STM development was based on the idea that moths integrate their pheromone detections over some finite period of time in order to make a maneuver "decision." This hypothesis has not appeared in the open literature. Experts in the field of pheromone-related moth behavior have not discounted this idea, however, more research needs to be conducted to verify its legitimacy. Having outperformed existing "moth-like" navigation algorithms (in trying to

match the MSexta wind tunnel tests) is notable achievement and warrents continued research as discussed in the next section. In addition to this contribution to the field of behavioral biology, the results of the simulations (of Chapter IV) show promising capabilities for tracking both 2-D and 3-D chemical plumes. Such success validates a solid contribution to the field of odor-based navigation.

As no 3-D navigation algorithm is present in the open literature, the Vertical Search routine developed for this thesis effort is a contribution the the field of odor-based navigation. The simplicity of this algorithm, which has potential to be expanded upon, does not diminish its usefulness. Playing a key role, in concert with the Backtrack routine, the Vertical Search routine enabled the UAV to relocate the plume once it was deemed lost. This is extremely important as the UAV is likely to lose contact with the plume more than once during the course of tracking a plume.

5.3 Recommendations

Given this research has defined new ground in both the development of a 3-D odor-based simulation environment and an autonomous odor-based tracking algorithm, a significant number of recommendations can be made. These include: additional simulation scenarios to be tested, modifications to be made to the existing simulation environment and/or navigation algorithm, implementation of a swarm of UAVS and robotic implementation.

The 3-D simulation studies of this thesis effort focused primarily on adjusting the values of ζ and ΔT_{ct}. As previously noted, the smallest values tested were the most successful regardless of plume type or density. These values should be lowered in hopes of finding the point of diminishing returns. These lower values should result in increased efficiency in locating the source. Again, this would be a result from the UAV spending less time outside of the plume. Along with changing these parameters, additional plumes should be tested (for both 2-D and 3-D algorithms). This will further validate the robustness of the navigation algorithms tested.

To further enhance the computer simulation environment, a recommendation is made to change the development environment. MATLAB® and SIMULINK® are not the most powerful software tools to use in developing computer simulations. Using C++ would be a better choice, potentially decreasing computation time and allowing for an easier interface with more graphically capable programming languages such as JAVA.

More evident modifications to the navigation algorithms could be tested. The STM fuzzy controller (both 2-D and 3-D), which dictates a new heading for the given aircraft, should be tuned. The fuzzy sets used in these controllers are designed using an ad hoc approach and are not altered during the research. Much can be gained by tuning these controllers. The length of the 3-D STM is one of the tunable parameters that could have a positive effect on the capabilities of the algorithm. Incorporating a long term memory with the purpose of mapping the plume is a plausible extension of the STM concept. Having the ability to map the plume leads to the UAV having the capability of estimating the plumes location upwind. This would lead the UAV to be more efficient in both the time to reach the source as well as fuel consumption.

The concept of only searching in the horizontal plane until losing contact with the plume (in the 3-D case) is likely hindering the current algorithm's success. The horizontal and vertical planes should be searched simultaneously. In discussions with Dr. Peter S. Maybeck, the idea of using helical search patters came to light. If the optimal search pattern for 2-D is of a sinusoidal form (as in the case of the moth), expanding the search to 3-D leads one to suggest the use of a helical search pattern. This would lead to changes in the Backtrack and Vertical Search routines, likely increasing their effectiveness/performance.

Using multiple UAVs to search for the location of a plume is an important recommendation. Trying to locate the source of a chemical plume by using a single UAV incorporating a navigation algorithm robust enough to track any plume structure it comes in contact with may not be the best solution to the problem. Having multiple

UAVs with varying capabilities, cooperating together to reach the source is a viable alternative. The UAVs could be designed to optimally navigate different plume structures or they could be designed with specific search capabilities, some designed to search vertical planes while others designed for horizontal planes.

As the 2-D and 3-D navigation algorithms performed well in the simulation studies (up to 90% success rate in both 2-D and 3-D), the next step is to design a robotic platform to incorporate the algorithms. The true capabilities of the algorithms will not be known until they have the opportunity to be tested against a real chemical plume. With numerous examples in the open literature developing a 2-D platform is not impossible. Incorporating the 3-D algorithm on a UAV, however, is more complex and without predecessors in the open literature. With the abundance of knowledge and capabilities the Air Force posses in the realm of UAVs, the extension of this simulation work to a hardware implementation is conceivable.

Appendix A. Fuzzy Logic Rules

This appendix gives the rule bases for all the fuzzy controllers developed in this thesis effort. These rules determine how inputs are mapped to outputs of a fuzzy controller. The 2-D rule bases are given first, followed by the 3-D rule bases.

A.1 2-D Navigation Algorithm

A.1.1 Tracking Rule Base. For this fuzzy controller the moth's velocity is always set to *Low*, therefore, omitted as an output for the list of rules:

1. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *-180*) then ($H_{M_{new_{rel}}}$ is *0*)
2. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *-135*) then ($H_{M_{new_{rel}}}$ is *45*)
3. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *-90*) then ($H_{M_{new_{rel}}}$ is *-45*)
4. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *-45*)
5. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *0*)
6. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *45*) then ($H_{M_{new_{rel}}}$ is *45*)
7. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *90*) then ($H_{M_{new_{rel}}}$ is *45*)
8. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *135*) then ($H_{M_{new_{rel}}}$ is *-45*)
9. If (STM_{ave} is *Entering the Plume*) and ($H_{M_{rel}}$ is *-180*) then ($H_{M_{new_{rel}}}$ is *0*)
10. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *0*)
11. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *-135*) then ($H_{M_{new_{rel}}}$ is *45*)
12. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *-90*) then ($H_{M_{new_{rel}}}$ is *-45*)
13. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *-45*)

14. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *0*)

15. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *45*) then ($H_{M_{new_{rel}}}$ is *45*)

16. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *90*) then ($H_{M_{new_{rel}}}$ is *45*)

17. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *135*) then ($H_{M_{new_{rel}}}$ is *-45*)

18. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{M_{rel}}$ is *180*) then ($H_{M_{new_{rel}}}$ is *0*)

19. If (STM_{ave} is *In the Plume*) and (T_Z is Long) and ($H_{M_{rel}}$ is *-180*) then ($H_{M_{new_{rel}}}$ is *45*)

20. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *-135*) then ($H_{M_{new_{rel}}}$ is *45*)

21. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *-90*) then ($H_{M_{new_{rel}}}$ is *-45*)

22. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *45*)

23. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *0*)

24. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *45*) then ($H_{M_{new_{rel}}}$ is *-45*)

25. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *90*) then ($H_{M_{new_{rel}}}$ is *-45*)

26. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *135*) then ($H_{M_{new_{rel}}}$ is *-45*)

27. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *180*) then ($H_{M_{new_{rel}}}$ is *-45*)
28. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *-180*) then (($H_{M_{new_{rel}}}$ is *0*)
29. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *-135*) then ($H_{M_{new_{rel}}}$ is *45*)
30. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *-90*) then ($H_{M_{new_{rel}}}$ is *45*)
31. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *45*)
32. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *45*)
33. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *45*) then ($H_{M_{new_{rel}}}$ is *-45*)
34. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *90*) then ($H_{M_{new_{rel}}}$ is *-45*)
35. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *135*) then ($H_{M_{new_{rel}}}$ is *-45*)
36. If (STM_{ave} is *Leaving the Plume*) and ($H_{M_{rel}}$ is *180*) then ($H_{M_{new_{rel}}}$ is *0*)

A.1.2 Searching Rule Base. For this fuzzy controller the moth's velocity is always set to *High*, therefore, omitted as an output for the list of rules:

1. If ($H_{M_{rel}}$ is *-180*) then ($H_{M_{new_{rel}}}$ is *90*)
2. If ($H_{M_{rel}}$ is *-135*) then ($H_{M_{new_{rel}}}$ is *90*)
3. If ($H_{M_{rel}}$ is *0*) then ($H_{M_{new_{rel}}}$ is *-90*)
4. If ($H_{M_{rel}}$ is *45*) then ($H_{M_{new_{rel}}}$ is *-90*)
5. If ($H_{M_{rel}}$ is *90*) then ($H_{M_{new_{rel}}}$ is *-90*)
6. If ($H_{M_{rel}}$ is *135*) then ($H_{M_{new_{rel}}}$ is *-90*)
7. If ($H_{M_{rel}}$ is *180*) then ($H_{M_{new_{rel}}}$ is *-90*)
8. If ($H_{M_{rel}}$ is *-90*) then ($H_{M_{new_{rel}}}$ is *90*)
9. If ($H_{M_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *90*)

A.1.3 Turn Rate Rule Base.

1. If ($H_{M_{delt}}$ is *0*) then (ω_M is *Straight*)
2. If ($H_{M_{delt}}$ is *45*) then (ω_M is *Slight Turn*)
3. If ($H_{M_{delt}}$ is *90*) then (ω_M is *Moderate Turn*)
4. If ($H_{M_{delt}}$ is *135*) then (ω_M is *Strong Turn*)
5. If ($H_{M_{delt}}$ is *180*) then (ω_M is *Strong Turn*)

A.2 3-D Navigation Algorithm

A.2.1 Tracking Rule Base.

1. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *-180*) then ($H_{UAV_{new_{rel}}}$ is *0*)
2. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *-135*) then ($H_{UAV_{new_{rel}}}$ is *45*)
3. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *-90*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
4. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *-45*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
5. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *0*) then ($H_{UAV_{new_{rel}}}$ is *0*)
6. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *45*) then ($H_{UAV_{new_{rel}}}$ is *45*)
7. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *90*) then ($H_{UAV_{new_{rel}}}$ is *45*)
8. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *135*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
9. If (STM_{ave} is *Entering the Plume*) and ($H_{UAV_{rel}}$ is *-180*) then ($H_{UAV_{new_{rel}}}$ is *0*)
10. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *0*) then ($H_{UAV_{new_{rel}}}$ is *0*)
11. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *-135*) then ($H_{UAV_{new_{rel}}}$ is *45*)
12. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *-90*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
13. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *-45*) then ($H_{M_{new_{rel}}}$ is *-45*)
14. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *0*) then ($H_{UAV_{new_{rel}}}$ is *0*)
15. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *45*) then ($H_{UAV_{new_{rel}}}$ is *45*)

16. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *90*) then ($H_{UAV_{new_{rel}}}$ is *45*)

17. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *135*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

18. If (STM_{ave} is *In the Plume*) and (T_Z is *Short*) and ($H_{UAV_{rel}}$ is *180*) then ($H_{UAV_{new_{rel}}}$ is *0*)

19. If (STM_{ave} is *In the Plume*) and (T_Z is Long) and ($H_{M_{rel}}$ is *-180*) then ($H_{M_{new_{rel}}}$ is *45*)

20. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *-135*) then ($H_{UAV_{new_{rel}}}$ is *45*)

21. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *-90*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

22. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *-45*) then ($H_{UAV_{new_{rel}}}$ is *45*)

23. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *0*) then ($H_{UAV_{new_{rel}}}$ is *0*)

24. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{M_{rel}}$ is *45*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

25. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *90*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

26. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *135*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

27. If (STM_{ave} is *In the Plume*) and (T_Z is *Long*) and ($H_{UAV_{rel}}$ is *180*) then ($H_{UAV_{new_{rel}}}$ is *-45*)

28. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *-180*) then (($H_{UAV_{new_{rel}}}$ is *0*)

29. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *-135*) then ($H_{UAV_{new_{rel}}}$ is *45*)

30. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *-90*) then ($H_{UAV_{new_{rel}}}$ is *45*)
31. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *-45*) then ($H_{UAV_{new_{rel}}}$ is *45*)
32. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *0*) then ($H_{UAV_{new_{rel}}}$ is *45*)
33. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *45*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
34. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *90*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
35. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *135*) then ($H_{UAV_{new_{rel}}}$ is *-45*)
36. If (STM_{ave} is *Leaving the Plume*) and ($H_{UAV_{rel}}$ is *180*) then ($H_{UAV_{new_{rel}}}$ is *0*)

A.2.2 Horizontal Search Rule Base.

1. If ($H_{UAV_{rel}}$ is *-180*) then ($H_{UAV_{new_{rel}}}$ is *90*)
2. If ($H_{UAV_{rel}}$ is *-135*) then ($H_{UAV_{new_{rel}}}$ is *90*)
3. If ($H_{UAV_{rel}}$ is *45*) then ($H_{UAV_{new_{rel}}}$ is *-90*)
4. If ($H_{UAV_{rel}}$ is *90*) then ($H_{UAV_{new_{rel}}}$ is *-90*)
5. If ($H_{UAV_{rel}}$ is *135*) then ($H_{UAV_{new_{rel}}}$ is *-90*)
6. If ($H_{UAV_{rel}}$ is *180*) then ($H_{UAV_{new_{rel}}}$ is *-90*)
7. If ($H_{UAV_{rel}}$ is *-90*) then ($H_{UAV_{new_{rel}}}$ is *90*)
8. If ($H_{UAV_{rel}}$ is *-45*) then ($H_{UAV_{new_{rel}}}$ is *90*)

A.2.3 Velocity Rule Base.

1. If ($H_{UAV_{delt}}$ is *0*) then ($V_{UAV_{new}}$ is *Fast*)
2. If ($H_{UAV_{delt}}$ is *45*) then ($V_{UAV_{new}}$ is *Medium Fast*)
3. If ($H_{UAV_{delt}}$ is *90*) then ($V_{UAV_{new}}$ is *Medium Slow*)
4. If ($H_{UAV_{delt}}$ is *135*) then ($V_{UAV_{new}}$ is *Slow*)
5. If ($H_{UAV_{delt}}$ is *180*) then ($V_{UAV_{new}}$ is *Slow*)

Bibliography

1. *The MATLAB Reference Guide*. The Math Works Inc., 24 Prime Park Way, Natick, MA 01760, 1992.

2. Arbas, E.A., M.A. Willis, and R. Kanzaki. *Organization of goal-oriented locomotion: pheromone-modulated flight behavior of moths*, volume Biological Neural Networks in Invertebrate Neuroethology and Robotics of *Neural Networks: Foundations to Applications*, chapter 8, 159–198. Harcourt Brace Jovanovich, 1993.

3. Baker, T.C. "Upwind flight and casting flight: complementary phasic and tonic systems used for location of sex pheromone sources by male moths". Doving K.B. (ed.) (editor), *Proceedings of the 10th International Symposium on Olfaction and Taste*, 18–25. Oslo, 1990.

4. Bau, J., K.A. Justus, and R.T. Cardé. "Antennal resolution of pulsed pheromone plumes in three moth species". *Journal of Insect Physiology*, 48(4):433–442, 2002.

5. Belanger, J.H. and E.A. Arbas. "Behavioral strategies underlying pheromone modulated flight in moths: lessons from simulation studies". *J Comp Physiol*, A:345–360, 1998.

6. Belanger, J.H. and M.A. Willis. "Adaptive control of odor-guided locomotion: behavioral flexibility as an antidote to environmental unpredictability". *Adaptive Behavior*, 4(3/4):217–253, 1996.

7. Cardé, R.T. and J.A. Farrell. "Chemical plume tracing: insects as model navigators", 2002. Grant Provided by Office of Naval Research.

8. Carvalho, J.C., M.T. Vilhena, and D.M. Moreira. "Comparison between Eulerian and Lagrangian semi-analytical models to simulate the pollutant dispersion in the PBL". *Applied Mathematical Modelling*, 2005.

9. David, C.T. *Mechanisms of directional flight in wind*, volume Payne T.L., Birch M.C., and Kennedy J.S., 49–57. Clarendon Press, Oxford, 1986.

10. Department of Biology, Philipps-University Marburg. "Neurobiology/Ethology". Http://cgi-host.uni-marburg.de/ omtierph/neuro/member.php?mem=stengl&lang=en(1 Mar. 2006).

11. Farrell, J.A., J. Murlis, X. Long, W. Li, and R.T. Cardé. "Filament-based atmospheric dispersion model to achieve short time-scale structure of odor plumes". *Environmental Fluid Mechanics*, 12:143–169, 2002.

12. Farrell, J.A., S. Pang, and W. Li. "Chemical plume tracing via an autonomous underwater vehicle". *IEEE J. of Ocean Engineering*, 30(2):428–442, 2005.

13. Holland, O. and C. Melhuish. *Some adaptive movements of animats with single symmetrical sensors*, volume 4, 55–64. MIT Press, 1996.

14. Hunter, M.W. *Reaction Based Grasp Force Assignment.* Ph.D. thesis, AFIT, 1996.

15. Hurley, R. "Lagrangian plume modelling in the atmosphere - mixing down of morning buoyant emissions". 11th Australasian Fluid Mechanics Conference, December 1992.

16. Ishida, H., G. Nakayama, T. Nakamoto, and T. Moriizumi. "Controlling a gas/odor plume-tracking robot based on transient responses of gas sensors". *IEEE Sensors Journal*, 5(3):537–545, June 2005.

17. Ishida, H., H. Tanaka, H. Taniguchi, and T. Moriizumi. "Mobile robot navigation using vision and olfaction to search for a gas/odor source". *Intelligent Robots and Systems*, 313–318. IEEE/RSJ, October 2004.

18. Kennedy, J.S. *Olfactory Responses to distant plants and other odor sources*, 67–91. Shorey H.H., and McKelvey J.J. Wiley-Interscience, New York, 1977.

19. Kennedy, J.S., A.R. Ludlow, and A.R. Sanders. "Guidance systems used in moth sex attraction". *Nature*, (288):474–477, 1980.

20. Kennedy, J.S., A.R. Ludlow, and A.R. Sanders. "Guidance of flying male moths by wind-borne sex pheromone". *Physiol. Entomol.*, (6):395–412, 1981.

21. Li, W., J.A. Farrell, and R.T. Cardé. "Tracking of fluid-advected odor plumes: strategies inspired by insect orientation to pheromone". *Adaptive Behavior*, 9(3-4):143–170, 2001.

22. Liebst, B.S. and C.H. Spenny. "Nonlinear Dynamic Model of the F-16 Aircraft", 2000.

23. Liu, Y.B. and K.F. Haynes. "Filamentous nature of pheromone plumes protects integrity of signal from background chemical noise in cabbage looper moth, Trichoplusia ni." *Journal of Chemical Ecology*, 18:299–307, 1992.

24. Loutfi, A., S. Coradeschi, L. Karlsson, and M. Broxvall. "Putting olfaction into action: using an electronic nose on a multi-sensing robot". *Intelligent Robots and Systems*, 337–342. IEEE/RSJ, October 2004.

25. Ludlow, A.R. *Application of computer modelling to behavioral coordination.* Ph.D. thesis, University of London, 1984.

26. Mafra-Neto, A. and R.T. Cardé. "Dissection of the pheromone-modulated flight of moths using the single-pulse response as a template". *Experientia*, 52:373–379.

27. Mafra-Neto, A. and R.T. Cardé. "Fine scale structures of pheroomone plumes modulates upwind orientation of flying moths". *Nature*, 369:142–144, 1994.

28. Malcolm, A.L. and A. J. Manning. "Testing the skill of lagrangian despersion model at estimating primary and secondary particulates". *Atmospheric Environment*, 35:1677–1685, 2001.

29. Marques, L., U. Nunes, and A.T. Almeida. "Olfaction-based mobile robot navigation". *Thin Solid Films*, 418:51–58, 2002.

30. Nevitt, G.A. "Olfactory foraging by antartic procellariiform seabirds: life at high reynolds numbers". *Biological Bulletin*, 198:245–253, April 2000.

31. Peterson, H.G. and B.K. Lamb. *A Lagrangian approach to studying instantaneous plume dispersion and concentration fluctuations.* Technical report, Montana Tech of the University of Montana, March 1998.

32. Preiss, R. and E. Kramer. "Mechanism of pheromone orientation in flying moths". *Naturwissenschaften*, 73:555–557, 1986.

33. Preiss, R. and E. Kramer. *Pheromone-induced anemotaxis in simulated free flight*, 69–79. Clarendon Press, Oxford, 1986.

34. Pumir, A., B.I. Shraiman, and M. Chertkov. "Geometry of lagrangian dispersion in turbulence". *The American Physical Society*, 85(25):5324–5327, 2000.

35. Ross, J.R. *Fuzzy Logic With Engineering Applicatrions.* John Wiley and Sons Ltd, 2 edition, 2004.

36. Ross, T.J., J.M. Booker, and W.J. Parkinson. *Fuzzy Logic and Probability Applications: Bridging the Gap.* ASA-SIAM Series on Statistics and Applied Probability. SIAM, Philidelphia, ASA, Alexandria, VA, 2002.

37. Ruan, F. and D. McLaughlin. "An investigation of Eulerian-Lagrangian methods for solving heterogeneous advection-dominated transport problems". *Water Resources Research*, 35(8):2359–2373, August 1999.

38. Rumbo, E.R. and K.E. Kaissling. "Temporal resolution of odour pulses by three types of pheromone receptor cells in Antherae polyphemus". *Journal of Comparative Physiology*, A(165):281–295.

39. Schiffman, S.S., B. McLaughlin, G.G. Katul, and H.T. Nagle. "Eulerian-Lagrangian model for predicting odor dispersion using instrumental and human measurements". *Sensors and Actuators B*, 106:122–127, 2005.

40. Stevens, B.L. and F.L. Lewis. *Aircraft Control and Simulation.* John Wiley and Sons, Inc., 1992.

41. The University of Arizona. "The Manduca Project". HTTP://insected.arizona.edu/manduca/mand_adult.html, 19 November 2005.

42. Thistle, H.W., H. Peterson, G. Allwine, B. Lamb, T. Strand, E.H. Holsten, and P.J. Shea. "Surrogate pheromone plumes in three forst trunk spaces: composite statistics and case studies". *Forest Science*, 50(5):610–625, 2004.

43. Vabø, R., G. Huse, A. Fernö, T. Jørgenson, S. Løkkeborg, and G. Skaret. "Simulating search behaviour of fish towards bait". *ICES Journal of Marine Science*, 61:1224–1232, 2004.

44. Vickers, N.J. and T.C. Baker. "Reiterative responses to single strands of odor promote sustained upwind flight and odor location by moths". *Proceedings of the National Academy of Sciences of the United States of America*, 91, 5756–5760. 1994.

45. Vickers, N.J. and T.C. Baker. "Latencies of behavioral response to interception of filaments of sex pheromone and clean air influence flight track shape in Heliothis virescens (F) males". *Comparative Physiology*, 178:831–847, 1996.

46. Weil, J.C., P.P. Sullivan, and C.H. Moeng. "The use of large-eddy simulations in Lagrangian particle dispersion models". *American Meteorological Society*, 2877–2887, December 2004.

47. Willis, M.A. "Odor-guided navigation in walking and flying insects". 131–137. ION 60th Annual Meeting, June 2004.

48. Willis, M.A. and E.A. Arbas. "Flight muscle activity underlying pheromone modulated zigzagging flight in male moths, Manduca sexta". *Soc. Neurosci. Abstr.*, 17:1245, 1991.

49. Willis, M.A. and E.A. Arbas. "Odor-modulated upwind flight of the sphinx moth, Manduca Sexta". *J Comp Physiol*, A(169):427–440, 1991.

50. Willis, M.A. and E.A. Arbas. *Active behavior and reflexive responses: another perspective on odor-modulated locomotion, in Insect Pheromone Research New Directions*, chapter 27, 304 – 319. International Thomson Publishing, 1997.

51. Willis, M.A. and T.C. Baker. "Behaviour of flying oriental fruit moth males during approach to sex pheromone sources". *Physiological Entomology*, 19:61–69, 1994.

52. Willis, M.A., J. Murlis, and R.T. Cardé. "Pheromone-mediated upwind flight of male gypsy moths, Lymantria dispar, in a forest". *Physiological Entomology*, 16:507–521, 1991.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
23–03–2006	Master's Thesis	Sept 2005 — Mar 2006

4. TITLE AND SUBTITLE

Bio-Inspired, Odor-Based Navigation

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Maynard John Porter III, Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/06-48

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/SNRP
Attn: Dr. Mikel Miller
2241 Avionics Circle
WPAFB OH 45433-7765; Phone: (DSN) 785-6127 x4274

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The ability of moths to locate a member of the opposite sex, by tracking a wind-borne plume of odor molecules, is an amazing reality. Numerous scenarios exist where having this capability embedded into ground-based or aerial vehicles would be invaluable. The main crux of this thesis investigation is the development of a navigation algorithm which gives a UAV the ability to track a chemical plume to its source. Inspiration from the male moth's, in particular Manduca sexta, ability to successfully track a female's pheromone plume was used in the design of both 2-D and 3-D navigation algorithms. The algorithms were developed to guide autonomous vehicles to the source of a chemical plume.
The algorithms were implemented using a variety of fuzzy controllers and ad hoc engineering approaches. The fuzzy controller was developed to estimate the location of a vehicle relative to the plume: coming into the plume, in the plume, exiting the plume, or out of the plume. The 2-D algorithm had a 60% to 90% success rate in reaching the source while certain versions of 3-D algorithm had success rates from 50% to 100%.

15. SUBJECT TERMS

bio-inspired, plume tracking, odor-based navigation, navigation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Juan R. Vasquez, LtCol, USAF (ENG)
U	U	U	UU	184	19b. TELEPHONE NUMBER *(include area code)* DSN: 785-3636 x7231; Juan.Vasquez@afit.edu